Université du Québec
à Rimouski

CONCEPTION D'UN SYSTÈME DE RECONNAISSANCE OPTIQUE DE CARACTÈRES IMPRIMÉS ET MANUSCRITS SUR FORMULAIRES FÉDÉRAUX ET DOCUMENTS HISTORIQUES

Mémoire présenté

dans le cadre du programme de maîtrise en ingénierie

en vue de l'obtention du grade de maître ès sciences appliquées (M.Sc.A.)

PAR

© MARC-ANTOINE COULOMBE

Août 2024

Composition du jury :

Raef Cherif, président du jury, Université du Québec à Rimouski

Jean-François Méthot, directeur de recherche, Université du Québec à Rimouski

Yacine Yaddaden, membre externe, Université du Québec à Rimouski

Dépôt initial le 3 avril 2024 Dépôt final le 13 août 2024

UNIVERSITÉ DU QUÉBEC À RIMOUSKI
Service de la bibliothèque

Avertissement

REMERCIEMENTS

Tout d'abord, je tiens à exprimer ma plus sincère gratitude envers Monsieur Jean-François Méthot, ayant agi à titre de directeur de recherche, pour ses nombreux et précieux conseils tout au long du projet et sa disponibilité constante. Son professionnalisme, son souci du détail et son approche humaine représentent pour moi une inspiration qui dépasse le cadre du projet de recherche et qui influera assurément la suite de mon cheminement, autant au niveau académique, professionnel et personnel.

Je tiens aussi à remercier ma famille et plus particulièrement mes parents qui ont toujours fait tout ce qui était en leur pouvoir incluant d'innombrables sacrifices afin de m'encourager et me permettre d'atteindre mes objectifs et mes rêves. Sans leur soutien, ce présent projet de recherche n'aurait probablement jamais eu lieu.

Je suis aussi reconnaissant envers la Direction de l'imagerie et des opérations du Receveur général m'ayant permis de réaliser ce projet de recherche au sein de leur institution. Ceux-ci ont su m'encourager, m'écouter et m'accorder une totale confiance concernant la réalisation du projet. Cette relation de confiance a été pour moi un pilier pour me surpasser et surmonter les nombreux défis rencontrés.

Finalement, je souhaite remercier les membres du jury pour l'attention qu'ils ont porté à mon travail. J'adresse ainsi mes remerciements au directeur du jury Raef Cherif et au membre externe Yacine Yaddaden.

RÉSUMÉ

La préservation, la retranscription et l'accessibilité des documents manuscrits représentent un enjeu coûteux et complexe pour le gouvernement du Canada. De plus, les données pouvant être de nature confidentielle, l'utilisation de solutions privées représente un risque pour l'intégrité de l'information. Ce projet de recherche vise ainsi à développer un système de reconnaissance optique des caractères (OCR) sur des formulaires structurés puis de quantifier le taux d'exactitude et le temps de traitement requis afin d'établir une référence pour de futurs développements. Le taux d'exactitude visé doit être supérieur à 85% en comparaison à un opérateur humain et doit traiter un document de quinze cases en moins d'une minute pour être économiquement viable. Le projet doit aussi permettre de déterminer des pistes de solutions en vue de développer une solution plus complexe pour des documents non-structurés de nature historique.

Afin d'atteindre ces objectifs, les principales étapes d'un système de reconnaissance de caractères sont étudiées et décomposées : le prétraitement, la segmentation, l'extraction des caractéristiques, la classification d'images et le post-traitement. Des solutions telles que la méthode *ORB*, la détection de contours, la transformée en cosinus discrète (DCT), la transformée en ondelettes discrète (DWT), la transformée de Hough et 190 réseaux de neurones différents sont notamment utilisés afin de détecter la position du texte dans l'image, d'extraire les caractéristiques et réaliser la classification des caractères.

Sur un formulaire structuré avec une écriture typographique, le taux d'exactitude moyen après post-traitement est de 91,99% et il faut en moyenne 4,02 s pour traiter une case. Pour une écriture manuscrite, le taux d'exactitude après post-traitement est de 94,27% et il faut en moyenne 5,90 s pour traiter une case atteignant ainsi les objectifs fixés en termes de taux d'exactitude. Des améliorations restent cependant à apporter, notamment au niveau de l'écriture typographique, du seuillage ainsi que la segmentation de caractères collés pour améliorer l'exactitude. Au niveau du temps de traitement, la méthode proposée à l'aide de fenêtrage ainsi que le dictionnaire pour les prénoms sont des avenues moins prometteuses.

Au niveau des documents historiques non-structurés, ceux-ci ont seulement été abordés. Toutefois, les résultats obtenus pour les formulaires structurés permettent d'établir que les principaux défis se trouvent au niveau de la détection du texte, la correction de l'alignement et de l'inclinaison ainsi que dans la segmentation de l'écriture cursive. Une solution à base de réseau de propositions régionales (RPN), de réseau de neurones convolutifs (CNN) et de réseaux de neurones récurrents (RNN) est suggérée afin de pallier certaines faiblesses observées.

Mots clés : Intelligence artificielle, reconnaissance de caractères manuscrits, HCR, documents historiques, réseaux de neurones, OCR, reconnaissance optique de caractères.

ABSTRACT

The preservation, transcription, and accessibility of handwritten documents represent a costly and complex challenge for the government of Canada. Additionally, as the data may be of a confidential nature, the use of a private solution poses a risk to the integrity of the information. Therefore, this research project aims to develop an optical character recognition (OCR) system for structured forms and to quantify the accuracy rate and processing time required to establish a benchmark for future development. The targeted accuracy rate should exceed 85% compared to a human operator and should process a fifteen-box document in less than a minute to be economically viable. The project should also identify potential solutions for developing a more complex solution for unstructured documents of a historical nature.

To achieve these objectives, the main steps of a character recognition system are studied and broken down: preprocessing, segmentation, feature extraction, image classification, and post-processing. Solutions such as the *ORB* method, contour detection, discrete cosine transform (DCT), discrete wavelet transform (DWT), Hough transform, and 190 different neural networks are notably used to detect the position of text in the image, extract features, and perform character classification.

On a structured form with typewritten text, the average accuracy rate after post-processing is 91.99%, and it takes an average of 4.02 seconds to process a box. For handwritten text, the accuracy rate after post-processing is 94.27%, and it takes an average of 5.90 seconds to process a box, thus meeting the set accuracy objectives. However, improvements are still needed, particularly in terms of typewritten text, thresholding, and segmenting stuck characters to enhance accuracy. Regarding processing time, the proposed method using windowing and a dictionary for first names are less promising avenues.

Regarding unstructured historical documents, they have only been addressed. However, the results obtained for structured forms indicate that the main challenges lie in text detection, alignment, and tilt correction, as well as in the segmentation of cursive writing. A solution based on regional proposal networks (RPN), convolutional neural networks (CNN), and recurrent neural networks (RNN) is suggested to address some of the observed weaknesses.

Keywords: Artificial intelligence, Handwritten character recognition, HCR, Historical documents, Neural network, OCR, Optical character recognition.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, DES SIGLES ET DES ACRONYMES

CNN	« Convolutional neural network » (de l'anglais) – Réseau de neurones convolutif
CSID	Centre de solutions en imagerie de documents
DCT	« Discrete cosine transform » (de l'anglais) – Transformée en cosinus discrète
DFT	« Discrete Fourier transform » (de l'anglais) – Transformée de Fourier discrète
DIORG	Direction de l'imagerie et opérations du Receveur Général
DPI	« Dots per inch » (de l'anglais) – Points par pouce
DWT	« Discrete wavelet transform » (de l'anglais) – Transformée en ondelettes discrète
FFT	« Fast Fourier transform » (de l'anglais) – Transformée de Fourier rapide
GIN	Gestion de l'information numérisée
HMI	« Human-machine interface » (de l'anglais) – Interface homme-machine
IA	Intelligence artificielle
LSTM	« Long short-term memory » (de l'anglais) - Réseau récurrent à mémoire court et long terme
OCR	« Optical character recognition » (de l'anglais) – Reconnaissance optique de caractère

RNN « Recurrent neural network » (de l'anglais) - réseaux de neurones récurrents

RPN « Region Proposal Network » (de l'anglais) - Réseau de propositions régionales

SPAC Services publics et Approvisionnement Canada

UQAR Université du Québec à Rimouski

INTRODUCTION GÉNÉRALE

Encore aujourd'hui, le principal vecteur de la préservation et la diffusion du savoir demeure la communication écrite. Grâce à cette méthode, il est possible d'accéder à des documents permettant le partage de connaissances historiques ou scientifiques. À titre d'exemple, c'est plus de 300 000 thèses et mémoires qui sont conservés dans la collection de Bibliothèque et Archives Canada (Mont-Royal, 2023). C'est grâce à ces bibliothèques et ces collections qu'il est possible de propager les connaissances vers les générations futures.

Même si la diffusion des connaissances est l'objectif principal, la préservation et l'accès à ces textes demeurent la source de plusieurs problématiques. À titre d'exemple, parmi les 300 000 thèses et mémoires, seulement 50 000 sont disponibles en format numérique (Mont-Royal, 2023). C'est ainsi moins de 17% de ces documents qui sont facilement accessibles partout au Canada ce qui nuit à la propagation des connaissances. De plus, la numérisation possède elle-même plusieurs problèmes, notamment dû au texte prenant souvent une forme manuscrite, ce qui n'est pas propice à une recherche de texte par mots-clés. La conservation des documents en format papier demande aussi de grands milieux contrôlés. Cependant, même en milieu contrôlé, les documents se dégradent avec le temps, ce qui crée une course contre la montre afin d'éviter d'éventuelles pertes de connaissances.

Afin de diminuer les coûts liés à la conservation et faciliter l'accès, des efforts de recherche sont effectués pour développer des systèmes de reconnaissance de caractères. Ces systèmes, une fois jumelés avec un système de numérisation, permettraient de conserver facilement de grands volumes de documents sous format numérique. Cependant, il n'existe toujours aucune solution répondant aux besoins d'exactitude, de vitesse de traitement, de flexibilité et de cybersécurité pouvant s'appliquer à tous les documents fédéraux.

Mise en contexte

La sélection de ce projet de recherche se base principalement sur les différents besoins mentionnés lors de discussions avec la *Direction de l'imagerie et opérations du Receveur Général (DIORG)*. Cette organisation, située à Matane au Bas-St-Laurent, se trouve sous la supervision du ministère *Services publics et Approvisionnement Canada (SPAC)* du gouvernement du Canada et possède plusieurs mandats. Son premier mandat historique est en lien, comme son nom l'indique, avec les Opérations du Receveur Général, c'est-à-dire la réconciliation bancaire, la détection de fraude ou le service à la clientèle pour tous les chèques émis au nom du gouvernement. Plus récemment, la Division du traitement virtuel a été créée afin d'aider à la consolidation de la paye. Cependant, dans le cadre du projet actuel, celui-ci se concentre sur les activités en lien avec la *Gestion de l'information numérisée (GIN)* et le *Centre de solutions en imagerie de documents (CSID)*. Ces deux secteurs ont principalement comme mandat d'offrir une solution complète de services de numérisation pour les différents ministères fédéraux.

Figure 1. Façade de l'immeuble René-Tremblay servant présentement de bureaux pour la *DIORG*

Ce mandat s'inscrit dans un désir du gouvernement d'assurer la préservation et la consultation des documents à moindre coût. C'est dans cette optique que le *CSID* approche de nombreux ministères afin de présenter les différents services disponibles et ainsi convaincre les ministères de migrer vers le format numérique. De son côté, le *GIN* s'occupe de préparer les documents, de les numériser, de retranscrire les informations, de faire un contrôle qualité puis de transmettre le résultat numérisé au client. Pour donner un ordre de grandeur, c'est en moyenne plus de 30 millions d'images de documents qui sont numérisées par le *GIN* chaque année.

Toutefois, même si le GIN possède le matériel et l'espace nécessaires pour numériser de plus grands volumes, celui-ci s'avère limité par la main-d'œuvre requise et les coûts élevés reliés. Comme mentionné lors de l'introduction générale, l'un des principaux intérêts de la numérisation est de permettre de facilement conserver et obtenir les informations se trouvant dans les documents. Or, l'organisation numérise des documents provenant de plusieurs ministères différents, ce qui amène des variations au niveau de la nature, la structure et l'âge des documents. On peut ainsi retrouver des documents plus récents, comme des formulaires d'adhésion à des plans dentaires tels que l'on peut voir à l'annexe I ou encore des rapports d'environnement remontant jusqu'à il y a plus de 100 ans comme on peut voir à l'annexe II. Ainsi, si l'on souhaite obtenir les informations se trouvant dans les documents de manière numérique, il est actuellement nécessaire de réaliser une retranscription par une personne dédiée à cette tâche. Cette retranscription manuelle est coûteuse et lente, ce qui rend la numérisation moins intéressante pour les ministères, mais monopolise aussi de la main-d'œuvre qui pourrait être employée à d'autres rôles stratégiques.

Figure 2. Photo de l'étape de numérisation à l'aide d'un appareil *IntelliScan XDS* par la société *Exela Technologies*

En discutant de la situation avec l'organisation, il est possible de visualiser l'effet qu'aurait un développement technologique dans ce milieu en permettant des solutions plus abordables et plus rapides. En effet, cela favoriserait une accélération de la transition des documents vers le format numérique permettant ainsi une conservation à moindre coût et une consultation plus facile pour les citoyens canadiens.

Ce projet s'inspire aussi de précédentes recherches de solutions qu'avait tenté la *DIORG* en communiquant avec des entreprises privées telles que *IBM* et *Hyperscience*. Après avoir réalisé une série d'essais avec ces entreprises, les résultats obtenus ne répondaient pas entièrement aux besoins (exactitude, vitesse et facilité d'utilisation). À titre d'exemple, le logiciel *Datacap* proposé par *IBM* n'a obtenu, en moyenne, que 64,52% d'exactitude par case. Ainsi, il est nécessaire d'apporter en moyenne 2,41 corrections par case sur un formulaire structuré manuscrit comme celui à l'annexe I selon le rapport fournit par *IBM* ce qui n'est pas économiquement viable pour de la numérisation à grande échelle. De plus, plus de 200 modèles de formulaires sont requis pour utiliser le logiciel selon les premières

estimations présentées par *IBM*, la solution est donc difficile à utiliser et généraliser par la *DIORG*. Quant aux documents non structurés comme l'exemple présenté à l'annexe II, aucune entreprise n'a présenté ses résultats dû à la complexité du problème. Ainsi, les performances obtenues par ces deux entreprises lors des essais permettent de valider la pertinence du projet de recherche puisqu'un besoin est toujours présent et il n'existe pas à l'heure actuelle de solution satisfaisante. De plus, comme les logiciels testés jusqu'ici proviennent du domaine privé, peu de documentation est accessible concernant leur fonctionnement. Non seulement ce manque de documentation représente un enjeu au niveau de l'entretien et du développement, mais cela représente aussi un enjeu du point de vue de la cybersécurité. En effet, la numérisation réalisée à Matane contient un grand nombre de renseignements personnels sensibles (Noms, adresses, Numéro d'assurance social, etc.). C'est pourquoi, il est requis pour un ministère tel que *SPAC* de connaître les détails du fonctionnement du système pour s'assurer que celui-ci répond aux normes de cybersécurité à chacune des étapes afin d'éviter de possibles fuites de données compromettant la sécurité et la confiance des citoyens. Confier un tel mandat à une firme privée externe est ainsi souvent complexe et peu de ces firmes acceptent de dévoiler les détails de leur système. Le projet de recherche est ainsi une occasion de documenter un système de reconnaissance de caractères et ses composants le constituant avec plus de détails facilitant ainsi de futures recherches et développements pour des applications manipulant des renseignements confidentiels.

Deux groupes sont impliqués dans la réalisation de ce projet : la *DIORG* et l'*UQAR* (Université du Québec à Rimouski). La *DIORG* a comme rôle de fournir les différentes bases de données et le matériel nécessaires aux expérimentations et au développement du système de reconnaissance des caractères (*OCR*). L'*UQAR*, de son côté, s'occupe de la planification du projet et des expériences, de la réalisation et de l'interprétation des expériences ainsi que du développement du système et de son implantation finale. La problématique, les objectifs, la méthodologie et l'organisation du mémoire sont explicités lors des sections ci-dessous.

Problématique

Dans le cadre de ce projet, aucune solution complète n'existe afin de répondre entièrement et adéquatement au contexte rencontré par la *DIORG*. En effet, les entreprises ayant été approchées au préalable par la *DIORG* ne sont pas arrivées à réaliser une reconnaissance optique de caractères manuscrits jugée satisfaisante. Quant à la littérature, celle-ci ne se limite qu'à des sections indépendantes d'un système plutôt qu'à son étude globale. Peu de données sont ainsi disponibles afin d'évaluer les réelles performances en situation concrète pour un système allant de la numérisation jusqu'à la détection d'erreurs de la reconnaissance. Afin de mieux comprendre la nature de ce défi, celui-ci a été décomposé en plusieurs éléments de problématique :

- **Le volume de documents et temps de traitement :** La littérature scientifique n'aborde généralement peu voire aucunement le temps nécessaire à la reconnaissance de caractères pour les solutions proposées. Or, dans le cadre de l'application requise pour la *DIORG*, ce paramètre est essentiel pour assurer la rentabilité et la viabilité du projet. Un système lent n'aurait pas les capacités de traiter le volume de documents manipulés à la *DIORG* en un délai raisonnable pour ses clients rendant ainsi ce système peu intéressant en comparaison à une manipulation par un opérateur.

- **La variabilité des documents :** La variabilité des documents rend les solutions disponibles sur le marché peu adaptées. En effet, la *DIORG* numérise des documents pour l'ensemble du gouvernement du Canada, la nature des documents peut être radicalement différente. Comme il est possible de voir aux annexes I et II, certains documents prennent une forme que l'on définit comme étant « structurée », c'est-à-dire des formulaires avec une structure fixe ou une forme « non structurée », c'est-à-dire des documents avec une structure variable d'un document à l'autre et dont on ne possède aucun a priori. Ces deux formes offrent chacune leur lot de défis qui rendent les solutions commerciales

inadaptées. En effet, un formulaire structuré nécessite typiquement de prédéfinir manuellement chacune des cases ou parties pour chacun des formulaires, ce qui devient rapidement complexe dans un écosystème fermé lorsque l'on souhaite avoir plusieurs centaines de formulaires. Pour un formulaire non structuré, aucune solution n'est tout simplement offerte commercialement à cause de la complexité du problème à localiser le texte dans des documents différents ainsi que la gestion de l'écriture cursive qui complexifie le traitement.

- **La variabilité de l'écriture :** Dans le même ordre d'idée que pour la variabilité des documents, l'écriture varie elle aussi grandement selon plusieurs facteurs. Par exemple, l'écriture peut être soit typographique ou manuscrite et le système doit être en mesure de remplir les conditions pour les deux scénarios. Dans le cas des lettres manuscrites, le système doit être robuste à une écriture qui est variable d'une personne à l'autre et qui dépend aussi de la culture ou de la langue, du type de crayon utilisé, de la largeur des traits ou encore de la couleur. Comme le Canada est un pays officiellement bilingue, le système doit être en mesure de reconnaitre des mots et des caractéristiques propres aux deux langues telles que les diacritiques. De plus, l'écriture manuscrite peut être représentée sous forme script ou cursive rendant la segmentation des lettres plus difficile. Des lettres scriptes peuvent aussi se toucher involontairement ou se chevaucher ce qui complexifie la problématique. À tout cela s'ajoute aussi des problématiques avec la taille et la largeur des caractères et des espaces entre les lettres et les mots qui varient selon la personne, mais aussi selon la police d'écriture utilisée. Certains caractères peuvent même être représentés de manière semblable ou identique selon le cas comme le chiffre « 1 », ainsi que les lettres « i » et « l ». En combinant l'ensemble de ces facteurs, l'écriture peut être difficile à distinguer même pour un opérateur humain.

- **Taux de reconnaissance :** Les taux de reconnaissance obtenus par les systèmes commerciaux sont trop faibles pour être économiquement viables à grande échelle en comparaison à un opérateur humain. Comme mentionné précédemment, le logiciel *Datacap* proposé par *IBM* n'obtient par exemple qu'un taux de reconnaissance de 64.52%. Ainsi, une revérification humaine est obligatoire sur l'ensemble des documents afin d'apporter les corrections nécessaires, ce qui réduit grandement l'intérêt d'un tel système, surtout lorsque l'on doit manipuler des millions de documents.

- **Compromis exactitude et rapidité :** Comme mentionné précédemment, la littérature ne considère que très peu la rapidité dans leur étude. Ainsi, il n'y a que peu d'informations permettant d'optimiser le ratio entre le taux d'exactitude et la vitesse de traitement qui permet de maximiser la rentabilité et l'intérêt d'un système d'*OCR* pour un important volume.

- **Minimisation de l'intervention humaine et généralisation :** Les systèmes commerciaux nécessitent toujours plusieurs interventions humaines autant au niveau de la revérification des documents pour corriger les erreurs qu'à l'ajout de modèles prédéfinis (mauvaise généralisation) ce qui rend les solutions coûteuses et ralentit grandement le traitement.

Objectifs

Le but principal de ce projet est de concevoir un système de reconnaissance de caractères manuscrits sur des formulaires fédéraux et tester ses performances dans une mise en situation réelle pour une implémentation dans les processus de la *DIORG*, puis de proposer des pistes pour un développement sur des documents historiques. Comme mentionné lors de la section précédente, les problématiques sont principalement reliées au temps de traitement, à la variabilité des documents, à la variabilité de l'écriture, au taux de reconnaissance, à l'optimisation entre l'exactitude et la rapidité ainsi qu'à la minimisation des interventions humaines et une meilleure généralisation. Ce projet cherche ainsi à apporter une solution innovatrice en se référant aux différents avancements dans le domaine pour concevoir un système qui peut s'avérer à la fois efficace, rapide et facile d'entretien grâce à l'aide de l'UQAR. Les objectifs principaux sont les suivants :

1. Réaliser la conception du système de reconnaissance de caractères pour des formulaires fédéraux comme l'on peut voir à l'annexe I. Par définition, on considère un formulaire « structuré » comme un formulaire contenant des cases prédéfinies pour chacune des informations désirées (chiffres, lettres et/ou symboles communs). On peut typiquement retrouver deux types de cases pour ce type de formulaires : les cases « fortement structurées » et les cases « faiblement structurées ». Il est possible d'observer chacun de ces deux types dans l'exemple fourni à l'annexe I. Une case « fortement structurée » concerne les cases avec des sous-divisions individuelles pour chacun des caractères attendus facilitant ainsi le découpage et l'identification (exemple : Date de naissance). Une case « faiblement structurée » concerne plutôt les cases où l'on ne sépare pas individuellement chaque caractère, mais l'on connait tout de même la position approximative de la case et le contenu attendu (exemple : Ville/village). Ces cases propres aux formulaires structurés ont l'avantage d'exploiter leurs caractéristiques pour aider aux performances du système. Ainsi, pour un formulaire structuré, on peut définir des objectifs plus élevés

pour être économiquement viables. Au niveau de la vitesse de traitement, le système doit permettre un traitement de ces formulaires à un rythme d'une page (une combinaison variable de 15 cases fortement structurées et faiblement structurées) par minute. Au niveau du taux de reconnaissance, le système doit avoir un taux minimal de 85% sur les cases du document en comparaison avec un opérateur humain effectuant la même tâche et identifier clairement les cas incertains pour minimiser le nombre d'interventions humaines. De plus, l'ajout de nouveaux formulaires structurés doit être facile et rapide pour permettre de s'adapter rapidement à une grande variété de documents.

2. Proposer des pistes de solution pour la conception d'un système de reconnaissance de caractères pour des documents historiques comme l'on voit à l'annexe II. À la nuance de l'objectif précédent, ce type de document ne possède pas nécessairement une structure prédéfinie qui permettrait d'orienter le système. Ce système doit ainsi être en mesure de déterminer indépendamment du document la position de texte dans l'image numérisée. Pour ce type de document, la solution jugée économiquement viable est toujours à un rythme d'une page par minute, mais avec un taux de reconnaissance minimal de 75% sur les cases du document en comparaison avec un opérateur humain effectuant la même tâche et d'identifier clairement les cas incertains pour minimiser le nombre d'interventions humaines.

Il est aussi possible de mentionner d'autres objectifs secondaires qui sont plus spécifiques aux besoins de la *DIORG* qu'à la recherche en elle-même, mais qui demeurent essentiels pour assurer la pertinence du projet pour l'organisme :

1. Avoir un système complet incluant une interface pour l'entrée, le traitement et la sortie des données ;

2. Avoir un système facile d'opération pour une personne non spécialisée ;

3. Avoir un système donnant le niveau de confiance de la validité des résultats ;

4. Avoir un système avec une interface permettant de revérifier facilement les résultats avec le niveau de confiance le plus faible ;
5. Avoir un système qui s’adapte facilement à de nouveaux formulaires ou documents ;
6. Avoir un système évolutif permettant un autoapprentissage des formes d’écriture.

Hypothèses générales

Afin d'atteindre les objectifs mentionnés, quatre (4) hypothèses générales ont été posées lors de la réalisation du système permettant la reconnaissance de caractères sur des formulaires structurés :

1. Le modèle du formulaire est disponible ;
2. Il n'y a pas d'autres déformations ou distorsions optiques/géométriques dans l'image du formulaire outre l'alignement et l'inclinaison :
3. Le texte dans un formulaire se trouve à l'intérieur des espaces désignés ;
4. Les caractères peuvent être considérés comme des contours distincts.

Pour la première hypothèse, comme le système s'applique sur des formulaires structurés, on considère que le document possède un modèle prédéfini sur lequel on peut se reposer pour potentiellement simplifier la reconnaissance, améliorer l'exactitude et accélérer le traitement. Cette hypothèse simplificatrice permet ainsi de savoir potentiellement où se trouvent les régions d'intérêts dans l'image et du contenu attendu dans ces régions.

Pour la seconde hypothèse, celle-ci permet essentiellement de supposer que les documents numérisés à la *DIORG* sont représentatifs du document papier et que les possibles déformations ou distorsions ont été éliminées dans une étape précédente à la reconnaissance des caractères lors du contrôle qualité.

Pour la troisième hypothèse, celle-ci est complémentaire à la première. Comme il est posé que le formulaire possède un modèle et ainsi une structure, la troisième hypothèse complète cette information en supposant ici que cette structure est respectée par les gens remplissant le formulaire structuré. Il s'agit ainsi d'une hypothèse simplificatrice puisque l'on ne cherche pas l'information attendue sur l'ensemble de l'image, mais seulement sur la région délimitée par la structure.

La quatrième et dernière hypothèse s'applique afin de limiter le projet de recherche à de l'écriture de type scripte, permettant ainsi de simplifier la problématique principalement au niveau de la segmentation de l'écriture cursive. Après discussion avec la *DIORG*, il a été convenu que cette hypothèse était justifiée pour des formulaires structurés puisque l'écriture scripte y est favorisée, mais que cette hypothèse ne pouvait s'appliquer à des documents non-structurés.

Méthodologie

Dans l'objectif de concevoir un système de reconnaissance optique de caractères manuscrits sur des formulaires fédéraux et des documents historiques, la méthode ci-dessous a été élaborée en plusieurs étapes séquentielles. Cette approche globale est, dans l'ensemble, la même pour un formulaire structuré ou non structuré, mais adapté selon le cas et s'inspire essentiellement de la consultation d'autres travaux réalisés tels que les articles de Baviskar (2021), Cheriet (2007), Pandian (2011), Martinek (2020) et Povolotskiy (2019). Les différences seront explicitées lors de chacun des chapitres concernés. Il est aussi important de noter que chacune des étapes mentionnées ci-dessous a été élaborée et testée individuellement avant assemblage et que l'ordre n'est ici établi que par cohérence avec l'échéancier du projet.

1. La première étape de la méthodologie est de faire une revue de la littérature afin de vérifier l'état de l'art, diviser la problématique en sous-problématiques et identifier des pistes de solutions pour résoudre chacune des sous-problématiques. Les sous-problématiques sont entre autres le prétraitement, la segmentation et l'extraction des caractéristiques (généralisation), la classification (vitesse de traitement et taux de reconnaissance) et le post-traitement (taux de reconnaissance et minimisation de l'intervention humaine).

2. La seconde étape de la méthodologie vise à réaliser une première ébauche du sous-système de prétraitement, de segmentation et de l'extraction des caractéristiques à partir des pistes de solutions retenues lors de l'étape précédente. Des formulaires fédéraux et des documents historiques agiront à titre de référence afin de comparer les performances et la pertinence de chacune des méthodes sur le type de document concerné et ainsi retenir les méthodes les plus efficaces pour les différents aspects (alignement, filtrage, seuillage, etc.).

3. La classification des images des caractères individuels constitue la troisième étape de la méthodologie. Plusieurs essais et architectures de réseaux de

neurones seront expérimentés à cette étape afin de déterminer l'architecture et les hyperparamètres permettant un taux de reconnaissance maximal en un minimum de temps de calcul.

4. À partir des résultats obtenus à la troisième étape, ceux-ci peuvent être utilisés afin de retourner au découpage fonctionnel et réajuster les méthodes développées dans les étapes précédentes puis de développer le sous-système de post-traitement à l'étape 4. Cette étape permet donc d'évaluer la répartition et le type d'erreur que génère le sous-système d'extraction des caractéristiques et de classification, mais aussi de comparer l'effet de différentes méthodes de détection et de correction des erreurs afin d'évaluer les méthodes les plus efficaces.

5. La dernière étape pour conclure le projet consiste à assembler les sous-systèmes conçus lors des étapes précédentes afin de n'avoir qu'un seul système global pour les formulaires structurés. Cette étape permettra ainsi une correction des erreurs éventuelles qui auraient pu persister. Cette dernière étape permettra aussi d'évaluer les performances générales du système en considérant les contributions de chacun des sous-systèmes et ainsi vérifier si les objectifs ont été atteints.

À la fin de ces différentes étapes, un bilan sera présenté afin de souligner les performances réelles du système de reconnaissance, mais aussi d'identifier des pistes d'améliorations et des pistes de solutions pour les documents historiques.

Contributions

La contribution à la recherche scientifique de ce projet se trouve principalement à trois niveaux. Dans un premier temps, ce projet permet de documenter et d'évaluer les performances d'un système complet de reconnaissance optique de caractères. La documentation étant manquante ou incomplète pour la plupart des systèmes offerts sur le marché (voir chapitre 1), ce projet de recherche permet d'établir une référence de système pour une application à grande échelle en considérant la vitesse de traitement, la variabilité des documents, le taux d'exactitude et le nombre d'opérations par un opérateur lors du traitement.

Dans un deuxième temps, ce projet de recherche utilise une approche différente de la littérature pour réaliser l'extraction des caractéristiques. La caractérisation de l'image en fréquence et spatialement à l'aide de plusieurs transformées en plus de l'utilisation de l'image brute en parallèle n'a jamais été employée dans la littérature consultée.

Dans un troisième et dernier temps, la classification à l'aide de réseaux de neurones par la méthode « Un contre tous » n'a elle aussi jamais été employée dans la littérature consultée. Cependant, la méthode peut s'avérer intéressante dans une application à grande échelle afin de permettre un traitement en parallèle synchrone ou asynchrone des données.

Organisation du mémoire

Ce présent mémoire est divisé en quatre chapitres distincts excluant une introduction et une conclusion générales. L'introduction générale sert principalement à décrire le contexte de la recherche, la problématique rencontrée, les objectifs fixés après rencontre avec l'ensemble des parties ainsi que la méthodologie employée dans le cadre de la recherche.

Le premier chapitre consiste en une revue de littérature visant à détailler l'état de l'art et valider la pertinence de la problématique par la même occasion. Ainsi, ce chapitre permet de regrouper les informations en lien avec le prétraitement, la segmentation et l'extraction de caractéristiques, la classification d'images, le post-traitement ainsi que des exemples d'interfaces provenant de logiciels et systèmes accessibles sur le marché.

Le deuxième chapitre couvre l'intégralité de la méthodologie explicitée précédemment, mais appliquée aux spécificités des formulaires structurés. Ainsi, le chapitre contient les spécificités du prétraitement, de la segmentation et de l'extraction des caractéristiques sur des formulaires, de la classification d'images et le post-traitement. Ce chapitre permet ainsi de brosser un portrait réel du fonctionnement du système.

Le troisième chapitre se consacre à la présentation des résultats obtenus à partir du système présenté lors du chapitre deux afin de vérifier l'atteinte des objectifs et les différents points à améliorer.

Le quatrième chapitre se consacre aussi à l'intégralité de la méthodologie explicitée précédemment, mais cette fois appliquée aux documents non structurés. Ce chapitre permet ainsi de résumer les solutions envisageables pour ce type de formulaire en se référant aux apprentissages et résultats obtenus dans les chapitres deux et trois.

Pour conclure, la conclusion générale résume l'état final du projet et énonce diverses recommandations et pistes de solutions pour de nouveaux travaux.

CHAPITRE 1
GÉNÉRALITÉ SUR LES SYSTÈMES DE RECONNAISSANCE OPTIQUE DE CARACTÈRES ET ÉTAT DE L'ART

1.1 INTRODUCTION

Ce chapitre présente l'ensemble des concepts permettant la compréhension du projet de recherche à la suite d'une revue de la littérature. Les connaissances en lien avec le prétraitement, la segmentation et l'extraction des caractéristiques sont d'abord présentées, suivi des méthodes de classifications d'images et des méthodes de post-traitement. Ces différentes étapes ainsi que leur ordre ont été établis en se référant à certains travaux réalisés et mis en commun par Baviskar (2021). En effet, dans cette publication qui est une revue de littérature, plus de 83 articles en lien avec la reconnaissance de caractères sont étudiés et une procédure globale peut être identifiée. De plus, on retrouve sensiblement la même démarche dans d'autres documents consultés en lien avec la reconnaissance de caractères, mais avec des objectifs différents tels que Cheriet (2007), Pandian (2011), Martinek (2020) et Povolotskiy (2019) pour ne citer qu'eux. Finalement, des exemples de logiciels commerciaux sont présentés afin d'établir un comparatif entre ce projet de recherche avec ce que l'on peut trouver sur le marché.

1.2 PRETRAITEMENT, SEGMENTATION ET EXTRACTION DES CARACTERISTIQUES

Les premières étapes que l'on retrouve généralement dans un système de reconnaissance optique de caractères sont le prétraitement, la segmentation et l'extraction des caractéristiques et ces étapes dépendent principalement de l'état initial des images

(Baviskar, Ahirrao, Potdar, & Kotecha, 2021). Les différents détails en lien avec chacune de ces étapes sont explicités aux sous-sections ci-dessous.

1.2.1 Prétraitement

Le prétraitement est généralement composé de trois (3) grandes étapes avec un ordre plus ou moins variable selon la nature du problème : la correction de l'alignement et la correction de l'inclinaison, la réduction du bruit (filtrage) et le seuillage (Baviskar, Ahirrao, Potdar, & Kotecha, 2021).

1.2.1.1 Correction de l'alignement et de l'inclinaison

Plusieurs systèmes de reconnaissance incluent aussi une correction de l'alignement (Baviskar, Ahirrao, Potdar, & Kotecha, 2021). Cette correction s'avère nécessaire à la suite d'une numérisation puisque, que ce soit de manière automatique ou manuelle, un léger angle peut influencer la position du texte dans l'image et compliquer la reconnaissance de celui-ci (particulièrement avec des formulaires structurés où le positionnement est une information utile à la reconnaissance) (Baviskar, Ahirrao, Potdar, & Kotecha, 2021).

Plusieurs méthodes sont disponibles afin de corriger l'alignement. Entre autres, on retrouve des méthodes telles que l'analyse par projections, la transformée de Hough ou encore les transformées morphologiques (Baviskar, Ahirrao, Potdar, & Kotecha, 2021). Ces méthodes s'avèrent assez efficaces pour des formulaires non structurés, parce qu'ils n'utilisent pas d'a priori sur le document pour déterminer l'angle. Toutefois, ces méthodes sont plus sensibles aux erreurs en comparaison à des formulaires structurés pour lesquels on posséderait un modèle de référence (détaillé au paragraphe suivant).

Concernant la correction de l'inclinaison du texte, les méthodes présentées pour l'alignement du document s'appliquent aussi à cette échelle. Par exemple, la transformée de

Hough permet de détecter des lignes dans l'image. Ainsi, en détectant les lignes et en connaissant l'angle de chacune, il est possible de déterminer l'alignement et l'inclinaison selon le résultat désiré comme à la figure 3. Toutefois, certains algorithmes plus avancés se basant sur les premières lettres de chacune des lignes de textes sont de plus en plus populaires dû à leur capacité de généralisation supérieure (Kavallieratou, Likforman-Sulem, & Vasilopoulos, 2018; Wigington, et al., 2018). Ces différentes solutions devront être mises à l'essai afin de déterminer la solution la plus adaptée aux documents de la *DIORG*.

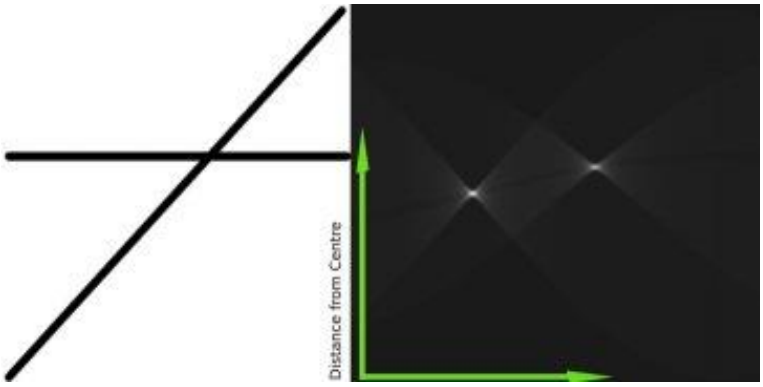

Figure 3. Exemple de détection de lignes à l'aide de la transformée de Hough (OpenCV, 2023) sous licence Apache 2.0

Certaines méthodes utilisent plutôt un document de référence (modèle) et effectuent une comparaison d'éléments clés pour déterminer l'orientation du document numérisé. Pour cette tâche, on retrouve principalement les algorithmes SIFT, SURF et *ORB* (Rublee, Rabaud, Konolige, & Bradski, 2011). Les algorithmes SIFT et SURF étant toutefois brevetés, ces solutions requièrent un coût et une complexité légale supplémentaires qui ne répondent pas aux besoins de la *DIORG*. Pour résumer très simplement l'idée derrière l'algorithme *ORB*, celui-ci détermine certains points similaires dans l'image avec une certaine confiance, conserve les points les plus pertinents selon un intervalle de confiance et calcule la matrice homographique permettant d'ajuster l'image au modèle comme il est possible d'observer à la figure 4. En effet, il est possible de calculer l'image corrigée ($[x' \; y']^T$) avec la matrice homographique (**H**) et l'image originale ($[x \; y]^T$) grâce à l'équation 1 :

Équation 1 : Transformation planaire homographique (OpenCV, 2023)

$$s\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{H}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{h_{11}} & \boldsymbol{h_{12}} & \boldsymbol{h_{13}} \\ \boldsymbol{h_{21}} & \boldsymbol{h_{22}} & \boldsymbol{h_{23}} \\ \boldsymbol{h_{31}} & \boldsymbol{h_{32}} & \boldsymbol{h_{33}} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Les performances de la méthode *ORB*, devront être comparées avec les méthodes sans modèle énumérées précédemment pour déterminer la solution la plus intéressante pour chacun des types de formulaires.

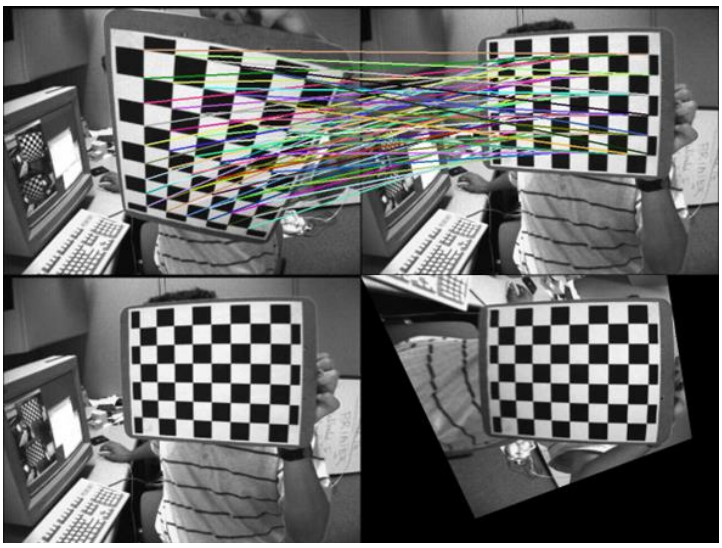

Figure 4. Exemple de détections de points clés et résultat après application de la matrice homographique (OpenCV, 2023) sous licence Apache 2.0

1.2.1.2 Filtrage

Selon l'image, une opération de filtrage peut s'appliquer afin d'ajouter un flou permettant d'accentuer des détails et des contours (voir équation 2) ou retirer le bruit en réalisant une convolution entre une matrice filtre et l'image (Szeliski, 2023). On peut distinguer deux types de filtres principalement : les filtres linéaires qui ont la particularité d'avoir une combinaison de poids fixes et les filtres non linéaires servant pour la morphologie (Szeliski, 2023).

Chacun des filtres que l'on peut voir à la figure 3.14 du livre de Szeliski donne un résultat différent et leur utilisation dépend du cas étudié. Les filtres carrés, bilinéaires et gaussiens sont des filtres analogues à des filtres passe-bas. Ceux-ci sont donc idéaux pour retirer du bruit haute-fréquence, mais entraînent un adoucissement de l'image perdant en netteté. Il est toutefois possible d'utiliser ces filtres pour accentuer l'image à partir de l'équation 2. Les filtres Sobel et détection de coins sont, quant à eux, analogues à des filtres passe-bandes. Ils permettent ainsi de détecter des arêtes dans l'image.

Équation 2. Équation pour accentuer une image à partir d'un filtre passe-bas (Szeliski, 2023)

$$g_{sharp} = f + \gamma(f - h_{blur} * f)$$

Concernant les filtres non linéaires, on peut souligner le filtre médian et le filtre bilatéral (Szeliski, 2023). L'une des principales utilisations dans le contexte de la reconnaissance de caractères est le retrait du bruit de type « sel et poivre » après numérisation ou d'un seuillage inadéquat que l'on peut retirer partiellement ou totalement à l'aide d'un filtre médian.

Ainsi, de la même manière que le seuillage, il n'existe pas de recette précise afin de filtrer adéquatement une image pour en améliorer la qualité. Il est donc nécessaire d'évaluer la combinaison adéquate lors de la mise en application du système après divers essais. La solution retenue sera présentée lors du chapitre 2.

1.2.1.3 Seuillage

Concernant le seuillage, cette étape est principalement réalisée pour faciliter la séparation entre le texte de premier plan et l'arrière-plan tel que décrit par Jyotsna (2016) ou pour réduire la quantité de données à manipuler pour les étapes suivantes comme mentionné

par Cheriet (2007). De cette manière, il est ainsi plus facile de réaliser la segmentation explicitée à la sous-section suivante. De plus, le seuillage possède aussi un intérêt au niveau de l'amélioration de la qualité de l'image, principalement par la réduction de bruit comme démontre Gupta (2007). Toutefois, les méthodes restent sensiblement les mêmes malgré les différents objectifs. De plus, en posant l'hypothèse qu'il n'est pas nécessairement avantageux de présenter des images d'écritures à niveau de gris à un réseau de neurones, le seuillage ne devrait pas nuire à la reconnaissance. C'est pourquoi, dans ce contexte, la reconnaissance peut être réalisée directement sur l'image binaire obtenue par le seuillage afin de profiter des avantages mentionnés pour la segmentation et la réduction du bruit. On retrouve principalement deux familles d'approches : le seuillage global et le seuillage local (Jyotsna, Chauhan, Sharma, & Doegar, 2016; Gupta, Jacobson, & Garcia, 2007; Cheriet, Kharma, Liu, & Suen, 2007).

Pour le seuillage global, la méthode consiste principalement à définir une valeur de seuil pour l'ensemble de l'image. Pour chacune des valeurs se trouvant sous ce seuil, le pixel considéré est mis à la valeur minimale (typiquement 0), alors qu'un pixel égal ou au-dessus de ce seuil est mis à la valeur maximale (typiquement 255 pour une image en format 8 bits entier non signé) (Szeliski, 2023). De cette manière, tous les pixels se retrouvent ainsi à la valeur minimale ou à la valeur maximale créant ainsi une image binaire généralement représentée par des pixels noirs (valeur minimale) et blancs (valeur maximale) comme l'on peut voir à l'équation 3. Plusieurs approches de seuillage global existent, la différence se trouvant principalement à la méthode de détermination du seuil (Cheriet, Kharma, Liu, & Suen, 2007). Les méthodes les plus couramment utilisées sont le seuil global fixe et la méthode d'Otsu décrite dans l'article ayant pour titre : *A threshold selection method from gray-level histograms* (Otsu, 1979).

Équation 3. Équation générale pour un seuillage à l'aide d'un seuil global (Szeliski, 2023)

$$\theta(f.t) = \begin{cases} 1 & Si\ f \geq t. \\ 0 & Sinon \end{cases}$$

Pour le seuillage local, cette approche subdivise l'image en régions de pixels et établit un seuil spécifique pour chacune d'entre elles (OpenCV, 2023). De manière équivalente, plusieurs méthodes existent avec comme principale différence la méthode de détermination du seuil local. Parmi les méthodes les plus populaires, on retrouve principalement les méthodes de Bernsen (1986), de Niblack (1986), de Niblack-Sauvola (Sauvola & Pietikäinen, 2000) ou encore de contraste adaptatif (Cheriet, Kharma, Liu, & Suen, 2007; Jyotsna, Chauhan, Sharma, & Doegar, 2016; Gupta, Jacobson, & Garcia, 2007). D'autres méthodes utilisent aussi une combinaison de seuils globaux et locaux pour obtenir des résultats équivalents. Cependant, comme le démontrent les résultats obtenus par Jyotsna (2016) et Gupta (2007), aucune approche, que ce soit avec un seuillage global, local ou une combinaison, ne permet d'obtenir un meilleur résultat dans tous les scénarios et ne dépend pas de l'image. Ainsi, ces différentes méthodes devront être évaluées pour le système de reconnaissance de caractères lors de la mise en application et détaillées dans le chapitre 2. À la figure 5 est présentée une comparaison du seuillage global et du seuillage local pour une image partiellement ombrée.

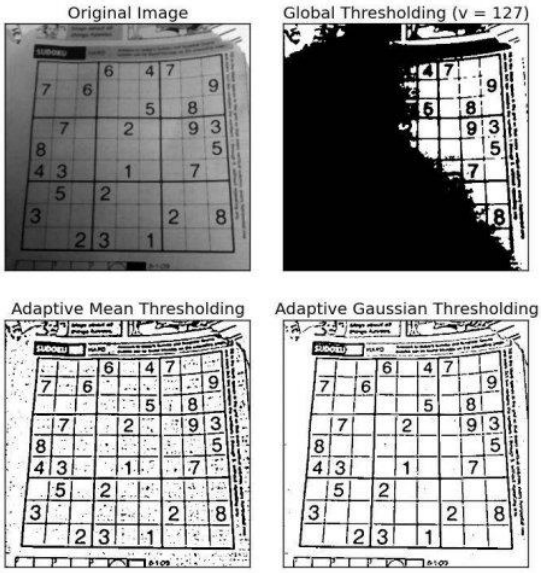

Figure 5 . Comparaison d'un seuillage global avec deux méthodes de seuillage local (OpenCV, 2023) sous licence Apache 2.0

Le schéma à la figure 6 permet de synthétiser les différentes étapes en lien avec le prétraitement de l'image.

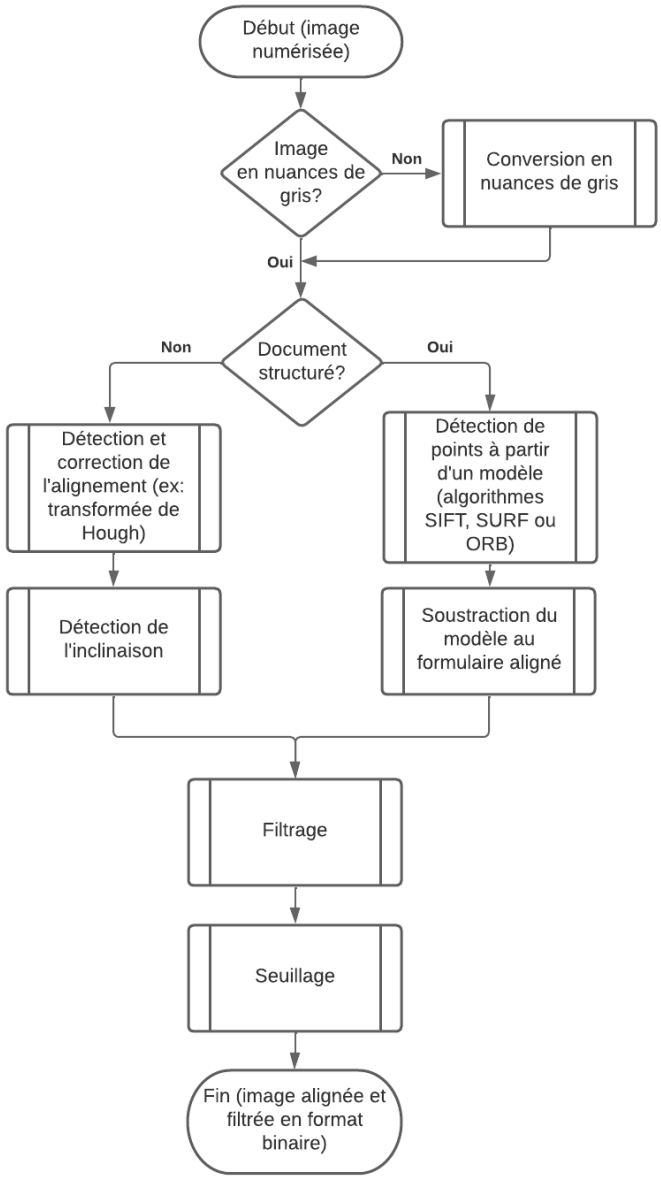

Figure 6. Schématisation du processus de prétraitement. L'entrée correspond à l'image du formulaire numérisé et la sortie permet d'obtenir une image alignée et filtrée du formulaire en format binaire. La conversion en nuances de gris n'a ici pas été explicitée comme la bibliothèque *OpenCV* offrait un résultat jugé satisfaisant et qu'un approfondissement aurait nécessité plus de temps et d'essais.

1.2.2 Segmentation

La segmentation permet, quant à elle, de découper les différents segments de texte sur le document permettant ainsi de découper les blocs de textes en lignes, en mots ou en lettres avant l'extraction des caractéristiques et la classification de l'image. On retrouve encore une fois plusieurs méthodes pour atteindre cet objectif.

Dans le cas d'un formulaire structuré, la segmentation s'avère plus simple grâce aux informations préalables sur le document. En effet, en connaissant la position des cases d'un formulaire et avec la correction de l'alignement et de l'inclinaison préalable, il est possible d'extraire directement les cases avec leur position absolue et d'extraire les contours fermés dans la case pour obtenir les lettres (Rosebrock, Thanki, Paul, & Haase, 2020).

Cependant, si une correction n'est pas effectuée au préalable ou si un modèle n'est pas disponible comme pour un formulaire non structuré, la segmentation s'avère plus complexe. La segmentation doit, pour ces cas, être dynamique. Ainsi, diverses méthodes doivent être appliquées comme notamment avec la transformée de Hough, la projection par histogrammes, les transformations morphologiques ou d'autres algorithmes créés à cet effet (Baviskar, Ahirrao, Potdar, & Kotecha, 2021; Martínek, Lenc, & Král, 2020; Povolotskiy & Tropin, 2019). Ces différentes méthodes dépendent encore une fois grandement de la nature du document. Des essais devront être réalisés afin de déterminer la solution la plus adéquate pour le système de reconnaissance de caractères que l'on souhaite concevoir.

La figure 7 ci-dessous permet de résumer l'ensemble des étapes à suivre afin de réaliser une segmentation du texte à partir de l'image alignée et filtrée.

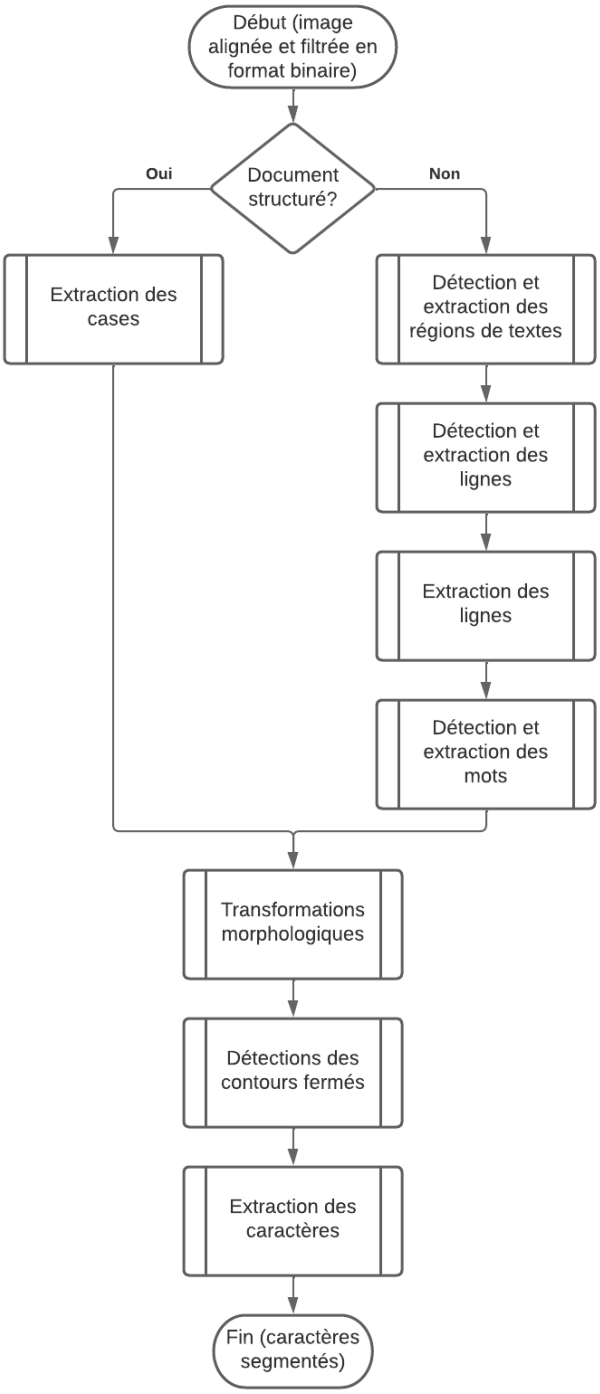

Figure 7. Schématisation du processus de segmentation. L'entrée correspond à l'image alignée et filtrée en format binaire et la sortie permet d'obtenir les images binaires isolées et centrées pour chacun des caractères (idéalement).

1.2.3 Extraction des caractéristiques

Une fois les lettres segmentées ou autrement dit, après avoir obtenu de petites images binaires (typiquement 28 pixels par 28 pixels) séparées pour chaque caractère pour correspondre au format de la base de données d'entrainement, il est alors possible d'extraire les caractéristiques de l'image afin de retenir les informations les plus pertinentes pour la classification d'images. On retrouve ici principalement deux approches pour caractériser les images : l'extraction avec spécification manuelle des paramètres et l'extraction automatique sans précision de paramètres.

L'extraction manuelle requiert, comme son nom l'indique, une étape d'extraction distincte qu'il faut programmer et ajuster manuellement selon les objectifs visés et le type de document. Une décision doit ainsi être prise lors du développement afin de sélectionner les caractéristiques qui permettront de mieux décrire et distinguer les caractères. Les principales caractéristiques étudiées pour les caractères sont les caractéristiques se basant sur des paramètres comme la longueur, le nombre de traits, les contours et la courbure pour décrire la structure (Memon, Sami, Khan, & Uddin, 2020; Rosyda & Purboyo, 2018). D'autres méthodes se basent sur des transformées ou des algorithmes afin de déterminer des caractéristiques sur l'espace ou la fréquence comme la transformée de Hough, la transformée de Fourier ou des transformations linéaires (Cheriet, Kharma, Liu, & Suen, 2007).

Quant à l'extraction automatique, celle-ci se base principalement sur les réseaux de neurones (Baviskar, Ahirrao, Potdar, & Kotecha, 2021). Cette approche consiste à laisser un réseau de neurones faire l'apprentissage à partir d'une base de données pour déterminer lui-même les caractéristiques les plus pertinentes (Szeliski, 2023; Bishop, 2006). De cette manière, il est possible de réaliser un seul réseau de neurones réalisant l'extraction des caractéristiques ainsi que la classification des images permettant donc de réduire le temps de développement et améliorer la généralisation, mais au coût d'un système prenant la forme d'une boîte noire complexifiant son analyse (Khan, Rahmani, Shah, & Bennamoun, 2018). En plus de pouvoir traiter directement l'image brute, le réseau de neurones permet aussi un

apprentissage sur des caractéristiques extraites selon les méthodes précédentes, comme par exemple, régénérer l'ordre de tracé comme caractéristique supplémentaire pour la classification (Rabhi, Elbaati, Hamdi, & Alimi, 2019).

Un résumé des principales approches utilisées par la littérature afin de réaliser l'extraction des caractéristiques est présenté à la figure 8 ci-dessous.

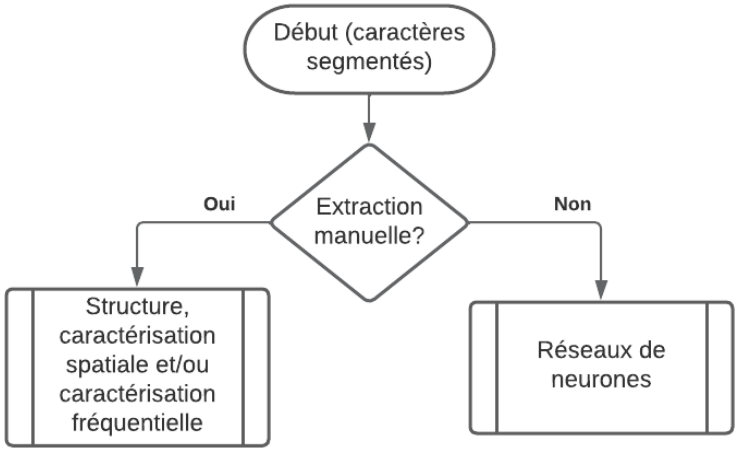

Figure 8. Résumé des principales approches pour l'extraction des caractéristiques. Par « caractères segmentés », on signifie ici des sous-images binaires de caractères idéalement isolés et centrés dans l'image.

1.3 CLASSIFICATION D'IMAGES

Concernant la classification de l'image, plusieurs approches peuvent être utilisées en combinaison avec l'extraction des caractéristiques. Comme mentionné précédemment, les réseaux de neurones sont une approche qui peut facilement se jumeler avec l'extraction des caractéristiques. Historiquement, on retrouve aussi cependant d'autres méthodes permettant d'obtenir d'excellents résultats tel les machines à vecteurs de support et les modèles de Markov (Baviskar, Ahirrao, Potdar, & Kotecha, 2021; Memon, Sami, Khan, & Uddin, 2020; Cheriet, Kharma, Liu, & Suen, 2007). Les réseaux de neurones demeurent toutefois la méthode la plus populaire dans la littérature scientifique de par leur performance comparable

aux autres méthodes tout en réduisant le nombre d'étapes de traitement en se combinant avec l'extraction des caractéristiques (Baviskar, Ahirrao, Potdar, & Kotecha, 2021; Memon, Sami, Khan, & Uddin, 2020). Un résumé est présenté à la figure 9 ci-dessous.

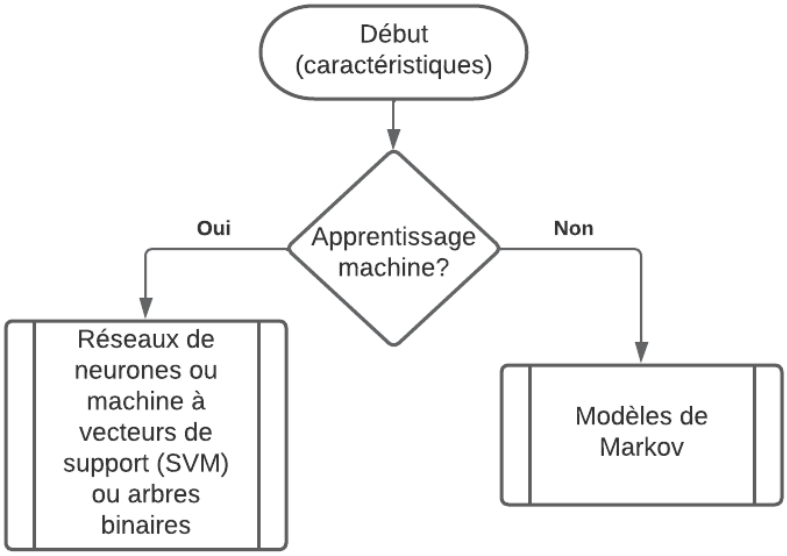

Figure 9. Résumé des principales approches pour la classification

1.4 POST-TRAITEMENT

Pour conclure les grandes étapes dans la conception d'un système de reconnaissance optique de caractères, on retrouve le post-traitement. Cette étape est cruciale pour améliorer le taux de reconnaissance global du système. En effet, le post-traitement a principalement pour objectif de détecter les erreurs résiduelles et de les corriger lorsque possible. Plusieurs méthodes de détection et de correction sont généralement utilisées pour ce faire et dépendent entièrement du type d'erreurs que l'on cherche à résoudre (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019).

Dans un premier temps, l'erreur peut être de nature humaine et aura des caractéristiques différentes des erreurs propres au système (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019). C'est-à-dire que l'erreur peut se trouver dans le document lui-même et n'est pas

induite par le système de reconnaissance. Dans ce genre de scénario, lorsque l'erreur est détectée et que l'on peut vérifier qu'il s'agit bien d'une erreur, la correction est favorisée afin d'éviter des retranscriptions erronées. Les erreurs de nature humaine ont la particularité de produire généralement des mots inexistants dans la langue de rédaction (67.5% des cas) (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019). Ainsi, l'utilisation d'un dictionnaire de mots est une solution qui est souvent employée pour corriger ce type d'erreurs puisqu'il permet de réduire le nombre d'erreurs de l'ordre de 40% (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019).

Cependant, l'utilisation d'un dictionnaire ne permet pas de considérer le contexte. Ainsi, un mot ayant une orthographe suffisamment proche peut remplacer le mot réel et ainsi ajouter une source d'erreurs supplémentaire. Pour minimiser l'impact de ce type d'erreurs, les solutions sont généralement de réduire la taille du dictionnaire, de favoriser les résultats les plus fréquents ou encore de considérer la probabilité de chaque type d'erreurs (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019). En effet, en limitant la taille du dictionnaire à une liste de mots possibles dans le contexte et en favorisant les mots les plus fréquents, il est possible de réduire les risques de remplacer une erreur avec un mot erroné. De la même manière, il est possible de réduire le nombre d'options possibles en considérant les probabilités des types d'erreurs dans un mot. Comme démontré dans l'article *Deep Statistical Analysis of OCR Errors for Effective Post-OCR Processing* (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019), les erreurs se trouvent le plus souvent au milieu du mot et se produisent rarement au début de celui-ci. Il est ainsi possible de statistiquement réduire les possibilités en se limitant aux mots ayant le moins de modifications requises, en considérant le type de modification (ajout, suppression ou remplacement) et en considérant la position des modifications dans le mot. D'ailleurs, certaines combinaisons de lettres ont plus de risques d'entraîner des erreurs, celles-ci peuvent être traitées avec une plus grande attention. À titre d'exemple, les lettres « l » et « i » peuvent être confondues, il peut parfois être favorable de ne pas faire la distinction lors de la classification et de considérer simultanément les deux options (Lukasik, et al., 2021).

En plus de risquer d'entraîner des erreurs, l'utilisation d'un dictionnaire de grande taille entraîne un grand temps de calcul afin de consulter l'entièreté de celui-ci pour chaque mot. Pour réduire ce phénomène sans réduire de manière trop importante le dictionnaire, il est possible d'utiliser un algorithme de migration auto-organisé (Nguyen, Le, Phan, & Zelinka, 2021). Toutefois, cette approche demeure récente et n'est que peu présente dans la littérature.

Un autre type d'erreur à considérer est une mauvaise segmentation des mots. En effet, autant pour un formulaire structuré qu'un formulaire non structuré, la segmentation peut être erronée ce qui ne peut être corrigé uniquement avec l'aide d'un dictionnaire. Pour corriger ce type d'erreur, l'approche standard lors du post-traitement est d'attribuer un jeton à chacune des détections et de considérer la longueur du jeton (incluant les espaces et symboles). De cette manière, on cherche si un mot existant pourrait être formé en combinant deux mots segmentés montrant une possible segmentation superflue. Ce type d'erreur est d'ailleurs la plus fréquente au niveau de la segmentation pour un système de reconnaissance de caractères (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019).

À partir de l'ensemble de l'information obtenue au sujet du post-traitement, la figure 10 ci-dessous permet de résumer les étapes qui composent ce volet.

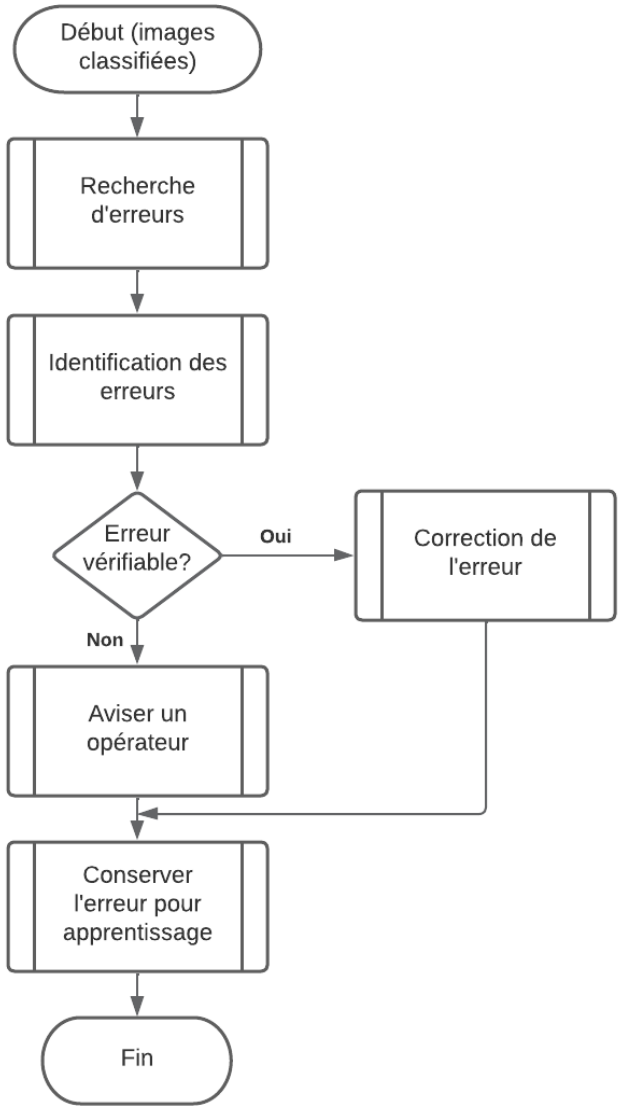

Figure 10. Schématisation du processus de post-traitement

1.5 EXEMPLES DE LOGICIELS COMMERCIAUX

Comme mentionné dans l'introduction générale, l'un des principaux systèmes de reconnaissance de caractères à grande échelle est le logiciel *Datacap* par *IBM*. Peu de documentation est accessible pour réellement décrire le fonctionnement et les performances

du système outre que celui-ci reçoit une image ou un PDF et réalise un traitement par intelligence artificielle (IBM, 2023). Il est ainsi difficile de caractériser le système dans son ensemble. Cependant, comme mentionné dans l'introduction, des essais ont déjà été réalisés avec l'entreprise et les résultats ne répondaient pas aux exigences de la *DIORG* dû aux faibles taux de reconnaissance obtenus.

De manière équivalente, le logiciel proposé par l'entreprise *Hyperscience* offre un système de reconnaissance de caractères similaire à celui offert par *IBM*. Le logiciel possède ainsi les problèmes équivalents, c'est-à-dire une documentation inaccessible ne permettant pas de connaitre le fonctionnement et d'établir les performances pour une application à grande échelle. La seule information connue sur le fonctionnement du logiciel est l'utilisation de l'apprentissage machine pour réaliser la reconnaissance (Hyperscience, 2023). Après des discussions entre l'entreprise et la *DIORG*, il n'y a encore une fois pas eu de suites dû à la difficulté à traiter des documents non structurés et le manque de documentation.

On retrouve aussi le logiciel *Tesseract* développé par *Google* sous forme de logiciel libre offrant ainsi plus d'informations quant à son fonctionnement. En effet, à partir de la documentation disponible, on remarque que le logiciel utilise, dans sa dernière version, un réseau de mémoire à long terme et donc un réseau de neurones pour réaliser l'extraction des caractéristiques et la classification (Tesseract-OCR, 2023). Toutefois, ce logiciel ne contient pas le prétraitement et le post-traitement, n'en faisant pas un système complet. De par ce fait, il est encore une fois difficile d'évaluer ce système pour de la reconnaissance de caractères à grande échelle et principalement concernant la vitesse de traitement. De plus, des exemples peuvent facilement être mis à l'essai démontrant une difficulté sur le texte manuscrit (Rosebrock, Thanki, Paul, & Haase, 2020).

D'autres logiciels disponibles comme *Adobe Acrobat Pro DC, OmniPage Ultimate, Abbyy FineReader, Readiris* ou *Rossum* sont plusieurs autres options envisagées, mais étant plus concentrées sur les particuliers plutôt que l'industrie. Ils ne sont ainsi pas adaptés aux besoins mentionnés dans l'introduction générale et ne possèdent encore une fois que très peu de documentation pour comprendre leur fonctionnement et quantifier leurs performances.

On ne remarque ainsi qu'aucune des options actuellement disponibles sur le marché ne permet de répondre aux critères de vitesse de traitement, de variabilité des documents et de taux d'exactitude tout en minimisant le nombre d'interventions par un opérateur. Dans les cas pouvant s'approcher de ces critères aux premiers abords, ceux-ci ne contiennent pas une documentation suffisante pour réellement évaluer si les performances sont atteintes comme il s'agit d'environnements fermés. Ces mêmes environnements fermés amènent aussi de grandes difficultés à comprendre le fonctionnement de ces systèmes et donc de le faire évoluer aux besoins. De plus, ce manque de connaissance introduit un risque important pour la sécurité comme il est souvent impossible de s'assurer que ces systèmes répondent aux normes de sécurités permettant d'assurer la confidentialité des données tout au long du processus.

1.6 CONCLUSION

Pour conclure, ce chapitre a permis de définir chacune des grandes étapes constituant un système de reconnaissance de caractères tels que le prétraitement, la segmentation, l'extraction des caractéristiques, la classification et le post-traitement. De cette manière, il est maintenant possible de comprendre qu'un système débute généralement avec une étape de prétraitement visant à seuiller l'image, la filtrer et corriger l'alignement et l'inclinaison facilitant la reconnaissance. Pour ce faire, plusieurs méthodes de seuillage peuvent être utilisées en définissant soit un seuil global pour l'image, la méthode Otsu étant l'une des plus populaires, ou encore avec un seuil local ou une combinaison des deux. Une étape de segmentation est ensuite réalisée pour détecter et séparer les différentes zones de textes et ainsi isoler des mots et des lettres. Chacune des lettres ou des mots subissent ensuite une extraction des caractéristiques dans le but d'obtenir les paramètres qui permettront de distinguer efficacement les caractères lors de la classification qui suit. L'extraction peut être réalisée manuellement à l'aide de la structure du caractère, de l'ordre du tracé et des

caractéristiques spatiales et fréquentielles de l'image ou de manière automatique à l'aide de réseaux neuronaux. Au niveau de la classification, les méthodes les plus populaires reposent principalement sur les avancements en apprentissage machine tels que les machines à vecteurs de support, les arbres binaires ou les réseaux de neurones. Toutefois, certains modèles utilisent aussi des modèles de Markov. Après avoir identifié les images lors de la classification, une étape de post-traitement est généralement nécessaire pour améliorer le taux d'exactitude en corrigeant les erreurs détectées de plusieurs manières. La méthode la plus courante est d'utiliser un dictionnaire puisqu'elle permet de réduire significativement le nombre d'erreurs, mais l'utilisation du contexte et l'attribution de jetons aux détections sont d'autres approches complémentaires. En combinant ces différentes étapes, il est alors possible de réaliser un schéma illustrant le processus d'un système global de reconnaissance des caractères. Un tel schéma est présenté à la figure 11 ci-dessous.

Finalement, les différents exemples de logiciels commerciaux tels que *Datacap* par *IBM*, la solution proposée par *Hyperscience* ou encore *Tesseract* par *Google* ont permis de vérifier qu'aucun logiciel ne répondait aux besoins ou ne fournissait une documentation complète du système et de ses performances validant la pertinence du projet de recherche. Lors du prochain chapitre, les solutions retenues et le fonctionnement du système sont présentés pour le système de reconnaissance optique de caractères sur des formulaires structurés et les performances finales sont évaluées.

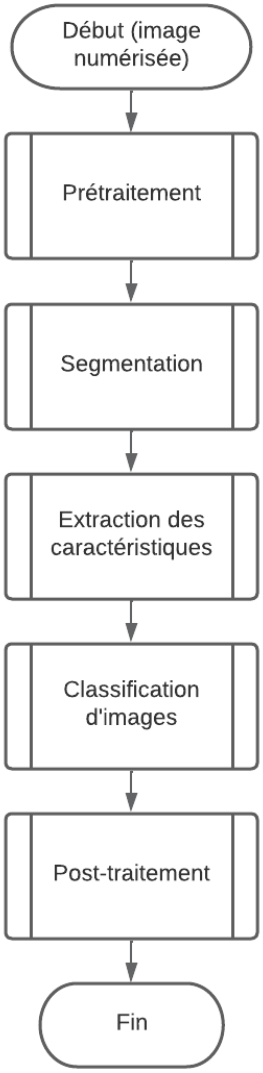

Figure 11. Schématisation du processus pour la reconnaissance de caractères

CHAPITRE 2
SYSTÈME DE RECONNAISSANCE SUR FORMULAIRES STRUCTURÉS

2.1 INTRODUCTION

Ce chapitre présente le fonctionnement du système de reconnaissance de caractères réalisé pour des formulaires structurés tels que celui présenté à l'annexe I. Ainsi, pour chacun des sous-systèmes identifiés lors du chapitre précédent à la figure 11, les solutions retenues sont présentées et détaillées permettant ainsi de brosser le portrait général complet du système et des différentes étapes nécessaires pour réaliser la reconnaissance de caractères à partir de l'image d'un formulaire. La programmation du système décrit se trouve, quant à elle, aux annexes V à XIII.

2.2 PRETRAITEMENT, SEGMENTATION ET EXTRACTION DES CARACTERISTIQUES

Les différentes étapes de prétraitement, de segmentation et d'extraction des caractéristiques qui ont été retenues à partir de la revue de littérature pour le système dans le cas de formulaires structurés seront présentées dans cette sous-section. La plupart des étapes de prétraitement réalisées se trouvent dans la programmation à l'annexe VI et ont été programmées à l'aide de la bibliothèque *OpenCV* (version 4.6.0 sous la licence Apache 2.0) en langage Python (Bradski, 2000). Une version équivalente a aussi été développée en langage C++ dans l'environnement Visual Studio 2019 pour le déploiement final à la *DIORG*.

2.2.1 Prétraitement

2.2.1.1 Correction de l'alignement et de l'inclinaison de la page

Pour la correction de l'alignement et de l'inclinaison, dans le cas de formulaires structurés, la méthode ayant été retenue est la méthode *ORB* (pairage de points ou de zones similaires entre deux images) puisque celle-ci donnait des résultats jugés satisfaisants après plusieurs essais et que cet algorithme n'est pas breveté comme les algorithmes SIFT et SURF simplifiant ainsi les procédures. Ces différents essais ont permis de déterminer qu'une détection de 5 000 points d'intérêts (et en n'en conservant 500) offrait des résultats satisfaisants pour le besoin actuel. Dans ce cas, l'erreur résiduelle est de l'ordre d'un à deux pixels sur l'alignement de la page complète. Ces paramètres pourront aussi être ajustés au besoin dans l'avenir afin d'améliorer les performances si jamais l'alignement et l'inclinaison deviennent des enjeux. Afin de sélectionner les points d'intérêts à conserver et ceux à rejeter, la distance est choisie comme critère de sélection. Ici, les meilleurs candidats retenus (500 points), représentent les combinaisons de points entre les deux images qui ont les caractéristiques les plus similaires et qui possèdent une distance minimale les séparant. Ce critère de sélection se base ici sur l'hypothèse que deux points similaires devraient se trouver sensiblement dans la même région du document et ainsi amener une droite horizontale contrairement à des points dans des régions éloignées qui amènent des droites obliques. Cette hypothèse n'est bien sûr pas valable pour l'entièreté des points, et certaines erreurs de pairage pourront subsister. Cependant, lorsque l'on retient les 500 meilleurs candidats, assez de candidats s'avèrent exacts pour assurer un alignement suffisant avec une erreur d'un à deux pixels pour les essais réalisés. À la figure 12 se trouve un exemple de points détectés entre un formulaire et son modèle.

Figure 12 : Exemple de détections et tracé de 5 000 points d'intérêts avec l'algorithme *ORB* sur un formulaire.

Une fois la matrice homographique calculée par l'algorithme *ORB*, le modèle est dilaté afin d'élargir les lignes noires du formulaire. De cette manière, en soustrayant le modèle au formulaire numérisé, il est possible de retirer tout ce qui est propre au modèle et ne conserver que les informations désirées. La dilatation du modèle permet ainsi d'éliminer partiellement l'erreur possible de quelques pixels sur l'alignement en surévaluant la largeur des traits. Un exemple de résultat obtenu est présenté lors de la sous-section traitant du filtrage à la figure 13. Cette étape de correction de l'alignement et de l'inclinaison est intéressante puisque comme l'on voit après filtrage obtenue à la figure 13, l'alignement à partir d'un modèle de formulaire vide avec l'algorithme ORB permet de prédéfinir des zones d'intérêts de formulaires à traiter (cases) afin de réduire la complexité du traitement.

2.2.1.2 Filtrage

Après différents essais sur quelques formulaires, deux types de filtrages ont été retenus pour la solution finale : le filtre médian et le filtre gaussien. L'utilisation de ces deux filtres survient lors de la soustraction du modèle et du formulaire avec l'algorithme *ORB*. Dans un premier temps, un filtre médian de dimension cinq par cinq est utilisé afin de retirer des pixels qui, lors de la soustraction, n'auraient pas été parfaitement alignés et ainsi éviter de générer du bruit supplémentaire provenant d'une erreur d'alignement. Le filtre gaussien de dimension treize par treize (dimension déterminée expérimentalement) est par la suite utilisé afin de réduire la taille de groupes de pixels qui auraient tout de même traversé le filtre médian, toujours afin de tenter de ne conserver que les pixels uniques au formulaire. Les coefficients σ_x et σ_y du filtre gaussien sont déterminés à partir de la taille du filtre avec la fonction « getGaussianKernel » de la bibliothèque *OpenCV*. Le résultat est présenté à la figure 13.

Figure 13 : Résultat de la soustraction du formulaire aligné avec le modèle après filtrage

2.2.1.3 Seuillage

Une fois que le formulaire a été aligné, que le modèle a été soustrait et que l'image a été filtrée afin de mettre en évidence les informations, il est alors possible de passer à l'étape du seuillage. Toutefois, en observant l'image à la figure 13, on remarque que plusieurs zones grises demeurent sur le formulaire et que l'intensité du texte varie à plusieurs endroits sur celui-ci selon la qualité de la conversion en nuances de gris, selon l'alignement, selon filtrage, selon l'épaisseur du crayon, etc. Cette observation a pu être validée en tentant d'utiliser un

seuil global sur l'image puisque celui-ci est difficile à ajuster et varie selon l'exemple de formulaire que l'on utilise. C'est pourquoi un seuil adaptatif local a été envisagé. Ceci est fait après avoir extrait chacune des cases du formulaire selon leurs coordonnées cartésiennes connues grâce au modèle, profitant ainsi de l'alignement du formulaire structuré pour faciliter le traitement. Après différents essais, la méthode Otsu permettait généralement un meilleur seuillage principalement dû au fait que cette méthode requiert le moins d'ajustement (s'adapte facilement à différents formulaires) et que la plupart des pixels ont un contraste assez important pour être efficacement segmentés à l'aide d'histogrammes.

Cependant, dans le scénario où une case du formulaire serait demeurée vide, la méthode Otsu a pour conséquence de surévaluer le seuillage parce que dans ce cas, l'histogramme est principalement composé de bruit. Ainsi, pour combler ce problème, un seuillage adaptatif de type gaussien est utilisé (largeur de quinze pixels) dans un premier temps. Ce seuillage et ces paramètres, obtenus après différents essais, ont principalement pour objectif de déterminer si des contours sont détectés dans l'image. Dans le cas où aucun contour n'est détecté après un seuillage adaptatif, on peut alors détecter une case vide et passer les différentes étapes suivantes. Ceci permet de diminuer le temps de calcul global. Lorsqu'un contour est détecté, peu importe la forme, on pose l'hypothèse que la case n'est pas vide et la méthode d'Otsu est utilisée sur l'image de la case pour obtenir un meilleur résultat et minimiser la faiblesse de cette méthode. Bien évidemment, dans l'ensemble les paramètres ont été ajustés de manière empirique, ceux-ci pourront être modifiés si des enjeux sont relevés au niveau du seuillage dans le futur.

2.2.2 Segmentation

Une fois le seuillage réalisé sur la case, le système tente ensuite de détecter la position du texte dans la case afin de réduire la présence de bruit et ne conserver que l'information pertinente à la reconnaissance. Pour ce faire, une série de dilations (en considérant une zone de cinq par cinq pixels) est réalisée. Ces nombreuses dilations établies expérimentalement ont pour but de fusionner les caractères se trouvant dans la case afin de ne former qu'un seul

groupe de pixels blancs tel que l'on peut voir à la figure 14, permettant ainsi d'obtenir une localisation du texte plus précise dans la case en retenant le plus grand contour et ainsi limiter la présence de bruit pouvant influencer la classification. Expérimentalement, il a été possible d'observer qu'un nombre de dilations entre 15 et 20 était adéquat pour les scénarios étudiés puisqu'ils permettent de bien joindre les différents caractères en un seul contour. Un nombre de dilatations plus faible amène un risque plus important à certains caractères d'être isolés et donc de ne pas être détectés dans la zone de texte. Un nombre de dilatations plus élevé a quant à lui le risque de surévaluer la région de texte, augmentant ainsi le risque d'ajouter du bruit. Lors des différents essais présentés lors du chapitre 3, celui-ci a été fixé à 17, mais pourra être ajusté de manière plus précise au besoin. Une fois le bloc de texte détecté avec la série de dilatation, le contour non dilaté est alors extrait du formulaire et une détection de contours est réalisée sur cette nouvelle région afin de segmenter les différents caractères se trouvant dans le bloc de texte. Un masque est finalement appliqué sur chacun des caractères détectés succinctement ce qui permet d'extraire le caractère tout en réduisant le chevauchement de caractères alentours et de redimensionner celui-ci en format 28x28 pour passer à l'étape de l'extraction des caractéristiques.

Figure 14 : Effet de dilations succinctes afin de joindre les caractères en un seul contour. L'image permet de comparer une case ayant après 0, 5, 10 et 15 dilatations successives.

2.2.2.1 Cas particuliers

La méthode présentée à la sous-section 2.2.2 fonctionne bien dans l'hypothèse où les caractères sont tous bien séparés et qu'aucun chevauchement n'est présent entre les différents

contours (quatrième hypothèse générale). Les caractères peuvent dans ce scénario être isolés individuellement et lus de gauche à droite. Dans ce scénario, il est alors possible de passer directement à la sous-section 2.2.3. Toutefois, certains cas particuliers demandent un traitement supplémentaire afin de permettre une meilleure segmentation soit lorsque deux caractères se chevauchent en hauteur et lorsque deux caractères sont collés et ne forment qu'un seul contour.

Lorsque deux contours se superposent en hauteur et que la distance centre à centre de ces contours est inférieure à un certain seuil, les deux contours sont fusionnés puisque l'on considère qu'il s'agit du même caractère malgré la séparation du trait (ex. : diacritiques) comme il est possible de voir à la figure 15. Expérimentalement, un seuil entre cinq et quinze pixels semblait être préférable. Un seuil inférieur à cinq augmente les risques de sur-segmentation alors qu'un seuil trop élevé risque d'entraîner une sous-segmentation. Lors des résultats présentés lors du chapitre 3, le seuil a été fixé à dix pixels, mais cette valeur peut être ajustée au besoin. Dans les autres cas, les contours sont considérés comme caractères distincts.

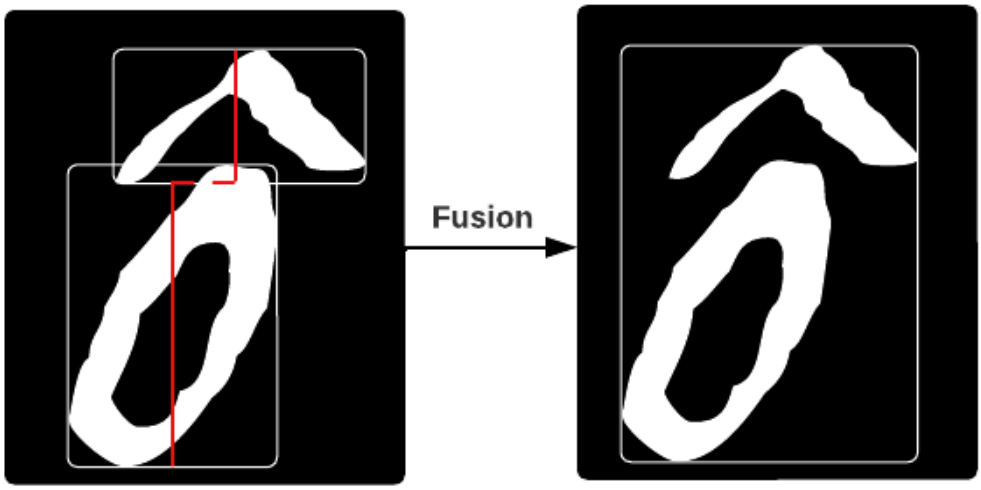

Figure 15 : Exemple de chevauchement entre deux contours pour lequel la distance centre à centre est inférieure au seuil fixé. Les deux contours sont fusionnés en un seul.

Dans le cas de figure où une classification lors du traitement des classifications individuelles (voir sous-section 2.3.1) résulterait en un poids de sortie du réseau de neurones sous un seuil désigné et que le contour est plus large qu'attendu, une seconde méthode de segmentation est utilisée afin de tenter d'améliorer le résultat. L'hypothèse ici choisie est de considérer qu'une faible certitude (indécision) combinée avec un large contour pourrait être la conséquence d'un contour contenant plus d'un caractère après une sous-segmentation. Expérimentalement, un poids de sortie du réseau de neurones inférieur à une valeur allant de 0,6 à 0,8 et une largeur de contours supérieur à 20 pixels semblent être une combinaison permettant d'identifier correctement un bon nombre de sous-segmentation. Cependant, cette méthode n'ayant pas été plus approfondie lors de la présente recherche, les paramètres ici suggérés seraient à analyser plus profondément.

Afin de séparer ces hypothétiques caractères doubles ou multiples, une fenêtre est glissée par pas de deux pixels le long du contour afin d'extraire des sous-images qui subissent ensuite une extraction des caractéristiques et une classification par des réseaux de neurones telles que décrites aux sections suivantes. Concernant la valeur du pas, des valeurs de zéro à quatre ont été testées et la valeur médiane a été retenue. Un pas plus faible a pour conséquence d'augmenter le temps de calculs, alors qu'une valeur plus élevée a pour conséquence de réduire la probabilité de trouver le bon caractère. La valeur du pas de deux semblait être le meilleur compris entre temps de calculs (deux fois plus rapides qu'un pas unitaire), tout en réduisant le risque de rater un caractère. Les coordonnées de chacune des sous-images et le poids de sortie du réseau de neurones associé sont retenus en mémoire jusqu'à ce que le balayage entier de la région soit complété. Une fois l'ensemble des poids obtenus, si une des sous-images obtient un poids de classification élevé à un certain seuil, on considère qu'un caractère a été reconnu dans cette sous-image et cette région de l'image est retirée temporairement. Il est alors possible de réaliser un nouveau balayage à gauche et/ou à droite de la région retirée afin de tenter de détecter d'autres caractères en suivant la même méthode, mais avec un second seuil. Concernant le seuil lors du premier balayage, il y a été possible d'observer expérimentalement qu'un seuil plus élevé est favorable afin de réduire le risque de faux positif qui entacherait l'erreur sur le second balayage. Ainsi, une valeur allant de 0,90

jusqu'à 0,98 semblait acceptable selon la tolérance aux erreurs. Pour le second seuil lors du second balayage, un seuil un peu plus faible semblait le plus adapté comme le premier caractère est considéré comme « certain ». Ainsi, une valeur allant de 0,7 jusqu'à 0,9 s'avère être un choix adéquat selon la tolérance aux erreurs. Ces deux paramètres ont été fixés à 0,95 pour le premier seuil et 0,8 pour le second pour les résultats obtenus lors du chapitre 3. Cependant, ces deux paramètres peuvent aussi être ajustés au besoin.

Lorsque la région à gauche ou à droite de la fenêtre n'est pas suffisamment large pour contenir un caractère, l'itération s'arrête et les caractères détectés sont retournés directement afin de passer au post-traitement. Si suffisamment de place demeure pour détecter un caractère, mais que la fenêtre est trop large ou qu'aucune détection n'a été réalisée au-dessus du seuil fixé, l'image est redimensionnée en réalisant une pyramide de l'image comme décrite dans l'œuvre de Kaehler et Bradski (2016) réduisant ainsi sa largeur de 30% et le processus de fenêtrage est répété. Le processus s'arrête ici lorsque l'image restante n'est plus de dimension suffisante pour contenir un caractère, que ce soit par le fait que l'ensemble de l'image ait été découpée en sous-images et retirée ou que le redimensionnement atteint une taille trop petite pour permettre de continuer. Un exemple simple du procédé est illustré à la figure 16. Cette méthode demeure toutefois une piste à approfondir dans une continuité du projet comme plusieurs paramètres ont été fixés temporairement à partir de quelques essais seulement et que celle-ci n'a pu être étudiée en détail afin de respecter l'échéancier.

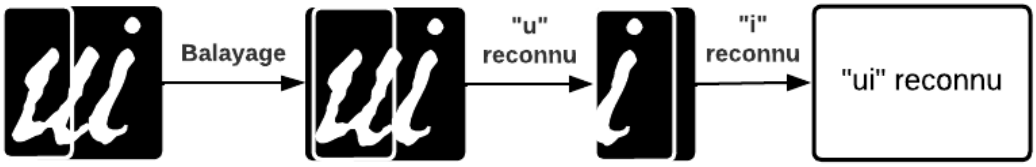

Figure 16 : Exemple de deux caractères qui se touchent. Segmentation à l'aide d'un balayage de fenêtre pour tenter de séparer les deux caractères. Cette façon de procéder ne fonctionne pas toujours et le travail reste à continuer.

2.2.3 Extraction des caractéristiques

Dans le cadre de ce projet de recherche, l'une des principales contributions concerne l'extraction des caractéristiques multiples et variées permettant de décrire chacun des caractères individuels à l'aide de trois transformées. En effet, la solution proposée après plusieurs expériences consiste à combiner les approches plus classiques telles que décrites par Cheriet (2007) ainsi que l'extraction automatique des caractéristiques à l'aide de réseaux neuronaux (Baviskar, Ahirrao, Potdar, & Kotecha, 2021). Pour ce faire, en plus de caractériser directement l'image brute à l'aide d'un réseau de neurones comme le fait la littérature, une caractérisation fréquentielle, une caractérisation spatiale ainsi qu'une caractérisation combinant fréquentielle et spatiale seront réalisées à partir des images brutes afin de considérer plus de paramètres lors de la classification et ainsi avoir une méthode plus robuste. L'objectif de cette approche est donc d'orienter la caractérisation des images afin d'obtenir différents apprentissages pour les réseaux neuronaux selon différentes caractéristiques et ainsi obtenir une meilleure généralisation du système. L'atteinte de cet objectif pourra être vérifiée lors de la présentation des résultats obtenus du système à la fin du chapitre 3. En plus de l'image brute, les différentes méthodes de caractérisations fréquentielles, spatiales et fréquentielles et spatiales sont présentées ci-dessous.

2.2.3.1 Caractérisation fréquentielle

Dans l'objectif de réaliser une caractérisation fréquentielle, la transformée de Fourier discrète (DFT) a d'abord été envisagée. Pour ce faire, les images de la base de données EMNIST ont été transformées en utilisant la fonction « *fft2* » de la bibliothèque *Numpy*. Cette fonction permet, entre autres, de calculer la transformée de Fourier discrète en deux dimensions à partir de l'algorithme de transformation de Fourier rapide (FFT) (Numpy, 2023). Afin d'obtenir un résultat plus visuel, la fonction «fftshift » de la même bibliothèque a aussi été utilisée pour décaler la fréquence nulle au centre de l'image. À la figure 17, un

exemple de transformation de l'image par la transformée de Fourier est présenté avec l'amplitude du spectre en échelle logarithmique (image centrale) et la phase (image de droite).

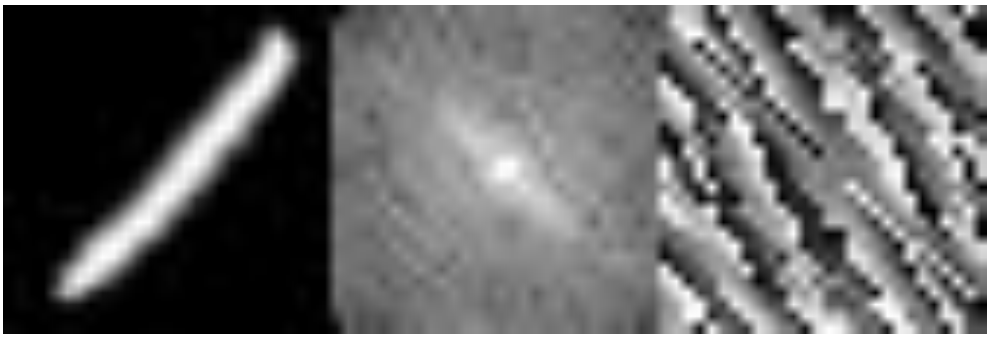

Figure 17. Représentation d'un « 1 » manuscrit avec la DFT : l'image brute à gauche, l'amplitude du spectre en échelle logarithmique au centre et la phase associée à droite.

Plusieurs essais d'apprentissage par réseaux de neurones ont été réalisés sur une image regroupant l'amplitude du spectre et sa phase. Toutefois, une seconde méthode a aussi été considérée afin de caractériser différemment l'image et déterminer la méthode la mieux adaptée. Ainsi, toujours pour caractériser l'image dans le domaine fréquentiel, la transformée en cosinus discrète (DCT) a été aussi envisagée. Cette transformée, principalement utilisée dans le domaine de la compression d'image, est un cas particulier de DFT n'employant que des termes réels (cosinus), mais d'approximativement le double de termes de la DFT permettant ainsi de la représenter en une seule image de plus grand taille (Ahmed, Natarajan, & Rao, 1974; Szeliski, 2023). À la figure 18, un exemple d'image obtenue en appliquant la DCT par la fonction « dct » de la bibliothèque *OpenCV* est présenté.

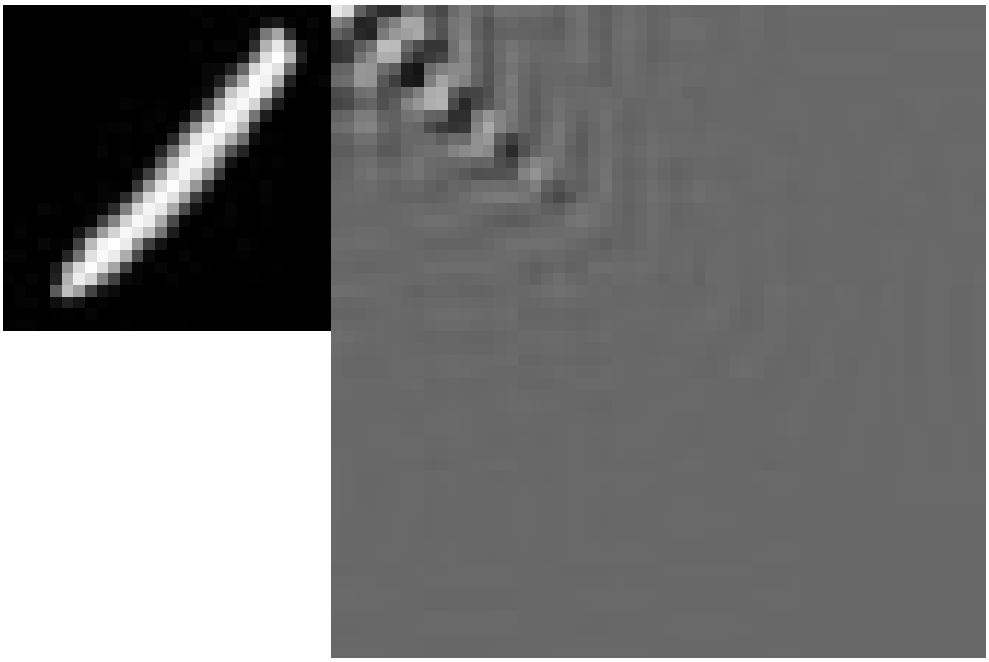

Figure 18. Représentation d'un « 1 » manuscrit : image brute à gauche et son équivalent obtenu en appliquant la transformée DCT à droite.

De la même manière que pour la DFT, plusieurs essais avec des paramètres identiques à ceux précédents ont été réalisés pour l'apprentissage par réseaux de neurones. Un extrait des résultats obtenus lors d'un essai est présenté au tableau 1.

Tableau 1 : Comparaison des performances d'apprentissage des caractéristiques fréquentielles sur des chiffres

Modèle	Nb. détections réussies	Nb. détections partielles	Nb. détections échouées	Nb. rejets	Taux de rejet (%)	Taux réussite minimal (%)	Taux de réussite moyen (%)	Taux réussite maximal (%)	Écart-type
DCT*	38904	147	46	903	2,26	99,4997	99,6943	99,8351	0,1019
DFT*	35575	497	159	3769	9,42	97,5081	98,8753	99,4980	0,6663

*Résultats provenant du répertoire d'expérimentation – Semaine du 2023-06-05

À partir du tableau 1, il est possible de constater que lors de cet essai, l'apprentissage sur les images ayant subi une DCT tend à mieux performer sur l'ensemble des critères établis.

En effet, on remarque un nombre de détections réussies et partielles (second résultat proposé permettant des suggestions à l'opérateur) beaucoup plus important, ce qui se répercute directement sur les taux de réussite (taux de reconnaissance). Le modèle DCT, en plus d'obtenir de meilleurs résultats, tend aussi à varier dans une plage beaucoup plus faible entre chacun des réseaux (écart-type six fois plus petit) ce qui crée moins de variation sur les performances du système. La sortie des réseaux utilisés dans ces tests étant une fonction Sigmoïde (détaillée à la section 2.3), la sortie de chacun de ces réseaux sont bornés de zéro à un. Une valeur étant plus près d'un « 1 » représentant ici une plus forte probabilité de présence du caractère, on peut alors rejeter les valeurs sous un certain seuil fixé arbitrairement selon la certitude et le taux de rejet toléré. Plus le seuil est fixé à une valeur élevée, plus le taux d'exactitude augmente au coût d'un plus grand nombre de rejets à revérifier par un opérateur. Plusieurs essais ont été réalisés dans le cadre du projet afin de déterminer expérimentalement un seuil (de 0,5 à 0,99) et des valeurs allant de 0,7 à 0,9 donnent généralement un taux de rejets acceptable tout en permettant de bons taux de réussite. C'est pourquoi un seuil de 0,8 est employé lors de cet essai mais cette valeur pourra être ajustée dans une étude plus approfondie selon la tolérance de la *DIORG*. Lors de cet essai, on remarque que le modèle DCT tend à attribuer un poids plus fort en sortie du réseau, ce qui s'exprime par un taux de rejet trois fois plus faible, ce qui est avantageux si l'on souhaite minimiser le nombre de vérifications par un opérateur. Afin de vérifier les tendances observées ici, des tests similaires ont été réalisés avec des paramètres d'apprentissage différents ainsi que sur des lettres. Toutefois, dans chacun de ces essais, des conclusions équivalentes sont obtenues.

Même si ces deux méthodes devraient théoriquement donner des résultats équivalents, la différence obtenue peut tout de même s'expliquer. Dans le cas de caractères manuscrits, la position est variable et l'image n'est pas toujours parfaitement centrée ou orientée de la même manière ce qui amène une certaine variabilité. Cependant, ces deux informations n'aident peu voire aucunement à identifier le caractère, c'est plutôt la forme brute du tracé peu importe sa position dans l'image qui permet de réellement reconnaître celui-ci. Dans le cas de la DFT, l'image est encodée en deux sous-images pour l'amplitude

et la phase. Or, dans ces deux images, celles-ci n'ont pas nécessairement une importance équivalente au niveau de la classification. En effet, comme la phase est beaucoup plus sensible à la position du caractère dans l'image, la sous-image de la phase sera sous-estimée par le réseau et des poids plus faibles lui seront attribués ce qui amène indirectement une sous-utilisation de l'information. En comparaison, la DCT permet de jumeler l'information de l'amplitude et de la phase en une seule image, forçant ainsi le réseau à utiliser une plus grande proportion de l'information qui lui est présentée ce qui permet, en théorie, une meilleure généralisation. Un autre facteur pouvant aussi influencer de manière importante le résultat est la taille des images avec les bibliothèques utilisées. Lors de la transformée DFT, le résultat obtenu est deux images de mêmes dimensions que l'image originale (28x28) permettant ainsi de générer une image (56x28). Cependant, dans le cas de la DCT, comme il s'agit d'une approche utilisant approximativement le double de termes en cosinus pour exprimer l'information au lieu d'une série de sinus et cosinus, l'image obtenue contient elle aussi plus de termes ce qui génère une image de plus grande dimension (56x56), encodant ainsi plus d'informations que la DFT dans le cas étudié et permettant encore une fois une meilleure généralisation grâce à ces informations supplémentaires pour décrire le caractère. La méthodologie retenue pour appliquer la transformée DCT après plusieurs essais est présentée à la figure 19.

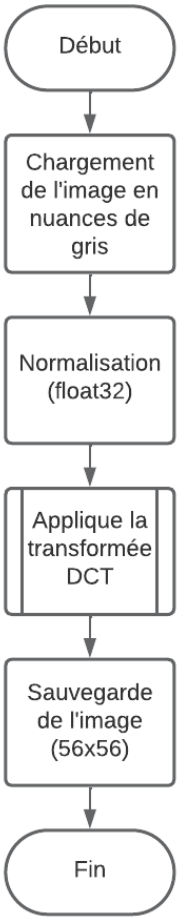

Figure 19. Schématisation du procédé d'application de la DCT. L'image obtenue est sauvegardée afin d'avoir une image de dimension uniforme de 56 pixels par 56 pixels.

2.2.3.2 Caractérisation spatiale

Au niveau de la caractérisation spatiale, une seule transformée a été retenue principalement dû à la prédominance dans la littérature comme il a été possible de constater lors du chapitre 1. La transformée de Hough étant documentée, fréquemment mentionnée dans la littérature et ajustable, celle-ci représente un candidat idéal. De la même manière que pour la caractérisation fréquentielle, la transformée de Hough a été appliquée sur l'ensemble des images de la base de données EMNIST pour n'en conserver que l'accumulateur.

Toutefois, comme l'accumulateur n'est généralement pas l'image désirée lors de l'application de la transformée, cette fonction a dû être programmée spécialement pour le projet. Un schéma résumant les grandes étapes de cette programmation après plusieurs ajustements se trouve à la figure 20. Un exemple d'image d'accumulateur obtenu par cet algorithme est aussi présenté à la figure 21.

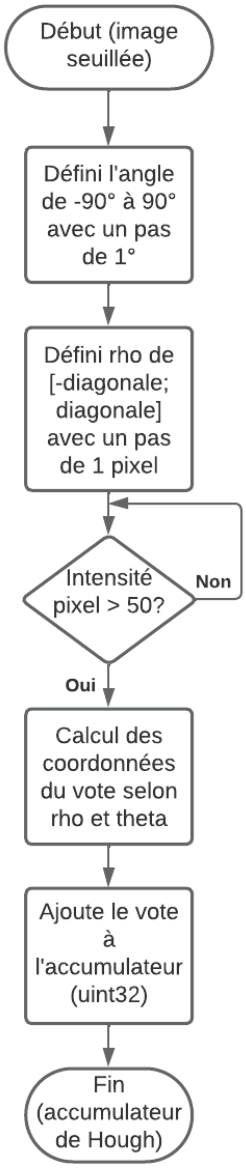

Figure 20. Schématisation de l'algorithme permettant de générer l'accumulateur de Hough

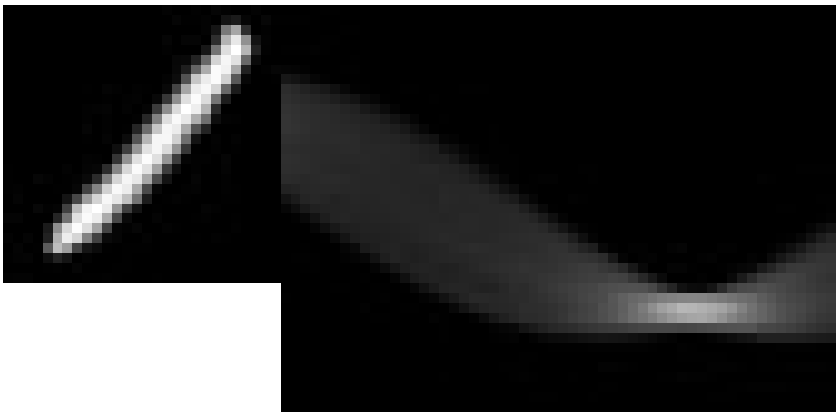

Figure 21. Représentation d'un « 1 » manuscrit : image brute à gauche et dans l'espace de Hough à droite. L'axe horizontal représente l'angle et l'axe verticale la distance rho.

Comme la transformée de Hough détecte des arêtes dans l'image, différentes approches ont été mises à l'essai pour déterminer la meilleure représentation dans l'espace de Hough pour l'apprentissage dans le scénario étudié. Parmi ces différents essais, le seuillage (à différents seuils), la squelettisation à l'aide de l'algorithme Zhang-Suen (1984) et un filtre de Canny ont été comparés. Il est possible de comparer un exemple des différentes images obtenues par ces essais aux figures 22 et 23.

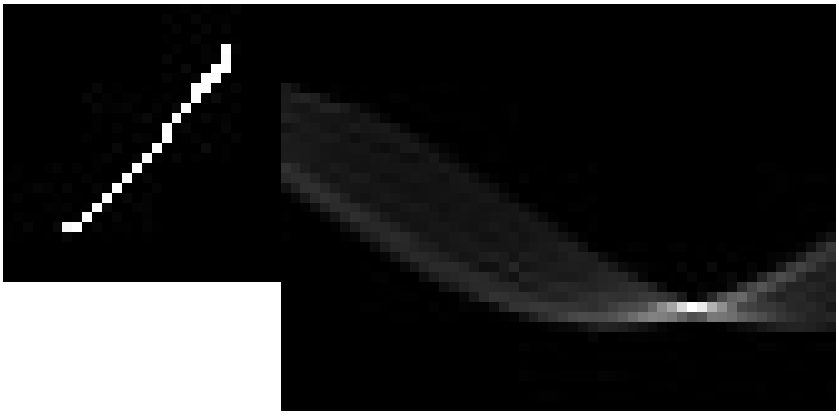

Figure 22. Représentation d'un « 1 » manuscrit squelettisé: image brute squelettisée à gauche et dans l'espace de Hough à droite

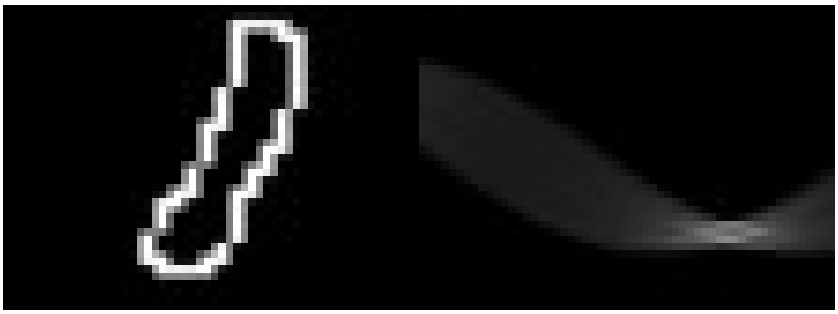

Figure 23. Représentation d'un « 1 » manuscrit avec détection de contours par Canny : image brute avec détection de contours à gauche et dans l'espace de Hough à droite

De manière équivalente à ce qui a été réalisé lors de la caractérisation fréquentielle, les différentes variations de la transformée de Hough ont été appliquées sur les chiffres manuscrits de la base de données EMNIST afin de comparer les performances pour différents paramètres d'apprentissage. Comme la transformée de Hough cherche à détecter des segments de droites, la méthode devrait théoriquement être plus performante sur des caractères avec des segments de droites et moins performantes sur les caractères avec des courbes. Un extrait de résultat se trouve au tableau 2 ci-dessous pour un seuil de 0,50 au niveau du réseau de neurones traitant les classifications individuelles (sous-section 2.3.2). Cet extrait ne contient que la transformée de Hough après un seuillage à cinq et cinquante sur les valeurs de l'accumulateur, mais durant la semaine d'expérimentation du 2023-07-10, des valeurs de seuillages allant de 5 à 100 par pas de 5 ont été évalués. Ce seuil permet ici de conserver ou d'éliminer des pixels de plus faibles intensités dans l'image lors du seuillage. Ainsi, en augmentant la valeur de ce seuil, on réduit le nombre de pixels considérés dans l'accumulateur de Hough et on ne retient que les principaux segments de droite dans l'image, mais en augmentant le seuil de manière trop importante, on risque de perdre de l'information pertinente. Au-delà d'un seuil de 50 et ce jusqu'à 100, l'augmentation du seuil n'amenait pas de changement distinctif dans les résultats outre la variance naturelle que l'on retrouve dans l'apprentissage avec des réseaux neuronaux. Toutefois, il était possible de remarquer une

amélioration distinctive progressive de 5 à 50 avec 50 comme valeur offrant les meilleurs résultats sur cette plage.

Tableau 2. Comparaison des performances des différentes variations de transformées de Hough sur des chiffres

Modèle	Nb. détections réussies	Nb. détections partielles	Nb. détections échouées	Nb. rejets	Taux de rejet (%)	Taux réussite minimal (%)	Taux de réussite moyen (%)	Taux réussite maximal (%)	Écart-type
Hough Seuil 5[1]	27437	179	44	319	1,14	99,2188	99,5174	99,8216	0,2157
Hough Seuil 50[2]	27589	147	27	216	0,77	99,1857	99,6380	99,9115	0,2237
Hough Squelettisé[2]	27346	252	73	308	1,10	98,6942	99,2808	99,8369	0,3242
Hough Canny[3]	26101	391	161	1326	4,74	96,6594	98,6624	99,5696	0,9550

1. Résultat provenant du répertoire d'expérimentation – Semaine du 2023-07-17
2. Résultat provenant du répertoire d'expérimentation – Semaine du 2023-08-21
3. Résultat provenant du répertoire d'expérimentation – Semaine du 2023-07-31

En comparant les différents résultats du tableau 2, il est possible de remarquer que la transformée de Hough après un seuillage global de 50 performe mieux sur l'ensemble des points, autant sur le nombre de détections réussies ou partielles, et possède une variation très faible entre les différents réseaux constituant ainsi le taux de rejet le plus faible nécessitant donc moins de revérifications de la part d'un opérateur. Encore une fois, ces résultats peuvent s'expliquer assez distinctement en regardant les images avant l'application de la transformée de Hough. Dans le cas d'un seuil bas, un nombre important de pixels passe à la valeur 1 lors du seuillage entraînant ainsi une forme moins distinctive dû à la présence de bruit qui est normalement éliminé pour un seuillage plus élevé. Quant à la squelettisation, le problème est principalement qu'en réalisant une telle opération afin de réduire à une largeur d'un seul pixel, une grande quantité d'informations est supprimée dans l'image rendant ainsi la caractérisation plus difficile. Pour la méthode avec le filtre de Canny, celle-ci ne tend en

réalité qu'à doubler la détection de lignes lors de la transformée de Hough, noyant ainsi l'information pertinente et rendant la transformée beaucoup moins nette comme l'on peut voir à la figure 23.

De plus, comme cette tendance se poursuit pour l'ensemble des essais réalisés sur les lettres malgré la modification des paramètres d'apprentissage lors des semaines suivantes, la transformée de Hough avec un seuillage global à 50 a été conservée pour la conception du système de reconnaissance de caractères. À la figure 24 se trouve une schématisation du processus retenu.

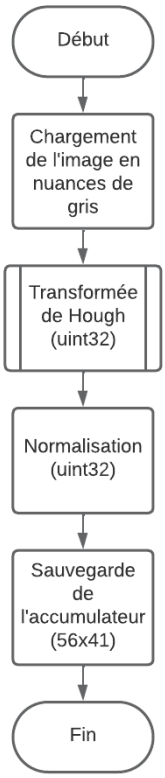

Figure 24. Schématisation du procédé d'application de la transformée de Hough. L'image de l'accumulateur est sauvegardée afin d'obtenir une dimension de 56 pixels par 41 pixels. Une partie des pixels en hauteur (15 pixels) ont été rognés comme étant toujours nuls puisque l'on peut exclure les lignes de 0 ou 1 pixel et que les lignes faisant l'entièreté de l'image sont impossibles puisqu'un contour vide est ajouté lors de la segmentation pour obtenir des images carrées

2.2.3.3 Caractérisation fréquentielle et spatiale

Jusqu'ici, les caractéristiques fréquentielles et spatiales ont pu être décrites individuellement à l'aide de la DCT ainsi que la transformée de Hough. Cependant, le fait de traiter ces deux approches séparément ne permet pas de caractériser la dépendance entre le domaine fréquentiel et le domaine spatial. Pour ce faire, une dernière transformée a été considérée, soit la transformée en ondelettes discrète (DWT) utilisée dans la compression de données pour le format JPEG2000 (Mekouar, 2001). La transformée par ondelettes permet ainsi de décomposer un signal (ici une image) à partir d'une fonction choisie afin de déterminer et localiser des irrégularités (spatiales) ou des variations (fréquentielles) permettant ainsi de caractériser simultanément les domaines (Mekouar, 2001). De manière générale, la transformée en ondelettes 2D dans le cas d'une image s'exprime par l'équation 4 suivante :

Équation 4. Équation de la transformée en ondelettes 2D d'une image (Damerval, 2008)

$$Wf(a.b.s) = \frac{1}{s^{2/p}} \int_{R^2} f(u.v)\psi\left(\frac{u-x}{s}.\frac{v-y}{s}\right) dudv$$

Dans cette équation, $f(u.v)$ représente l'image, ψ représente une fonction choisie arbitrairement (ondelette mère) qui répond à certaines conditions décrites par Damerval (2008) et s représente l'échelle. Dans le cas de cette recherche, l'ondelette mère s'est limitée à l'ondelette de Haar qui se définit comme :

Équation 5 : Équation de l'ondelette de Haar (Mekouar, 2001)

$$\Phi(x) = \begin{cases} 1 & si\ 0 \leq x \leq \frac{1}{2} \\ -1 & si\ \frac{1}{2} \leq x \leq 1 \\ 0 & ailleurs \end{cases}$$

Parmi les différents types d'ondelettes mères possibles, l'ondelette de Haar a été celle retenue pour ce projet principalement dû à sa forme à trois niveaux. En effet, comme les images sont déjà seuillées, une forme d'ondelette discrétisée semblait la plus appropriée pour décrire le contenu de l'image. Cependant, d'autres types d'ondelettes auraient pu aussi être mises à l'essai.

De par ces caractéristiques, la transformée en ondelettes permet une analyse à multirésolution permettant de diviser l'image en sous-images de différentes bandes de fréquences (Mekouar, 2001). Ces différentes sous-images s'obtiennent par le banc de filtres à la figure 25 et permettent, avec l'agencement à la figure 26 de créer l'image à la figure 27. Dans le cadre du projet de recherche, un seul étage de décomposition est utilisé même si un plus grand nombre d'étages est parfois utilisé dans la littérature comme dans l'article de Parida (2017). Ce choix s'explique principalement par la faible dimension des images utilisées (56x56). En effet, comme une décimation par deux est réalisée à chaque décomposition, la résolution tend à devenir trop faible pour contenir de l'information pertinente à la reconnaissance, limitant ainsi le nombre de filtrages possible.

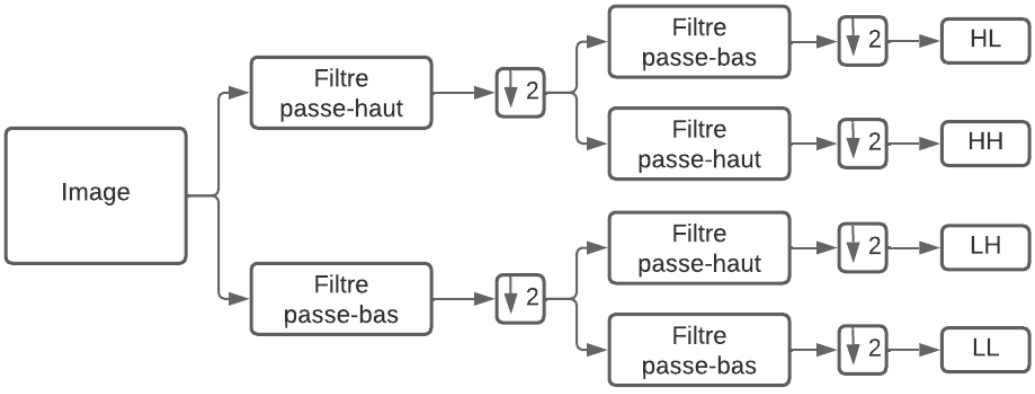

Figure 25. Décomposition d'une image à l'aide de la DWT (Parida & Bhoi, 2017; Mekouar, 2001). Les cases avec le symbole fléché et le chiffre 2 représente une décimation par deux (sous-échantillonnage).

À partir des différentes sous-images obtenues par la décomposition, une nouvelle image est constituée de la manière présentée à la figure 26. Cette image combine ainsi des caractéristiques fréquentielles et spatiales. Un exemple de résultat final réalisé sur un chiffre se trouve à la figure 27.

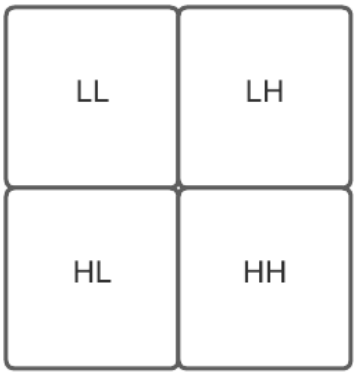

Figure 26. Image résultante de l'assemblage des sous-images de la décomposition

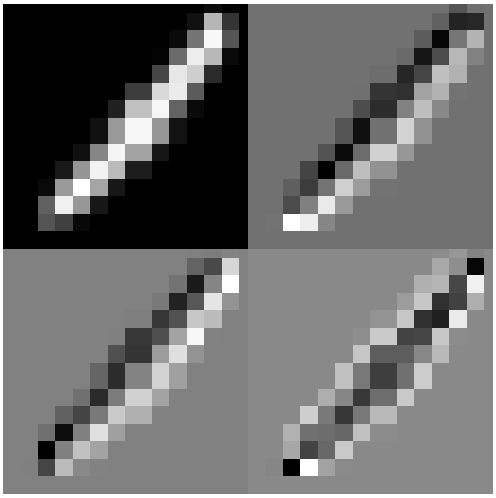

Figure 27. Représentation d'un « 1 » manuscrit décomposé par la DWT avec une ondelette de Haar. Dans le coin supérieur gauche, la décomposition à double filtre passe-bas. Dans le coin supérieur droit, la décomposition à un filtre passe-bas suivi d'un filtre passe-haut. Dans le coin inférieur gauche, la décomposition à un filtre passe-haut suivi d'un filtre passe-bas. Dans le coin inférieur droite, la décomposition à deux filtres passe-haut

2.3 CLASSIFICATION D'IMAGES

L'architecture du système retenu pour réaliser la classification des images à partir des caractéristiques extraites lors de la sous-section précédent sera présentée. La classification individuelle des caractères qui prend une approche différente de la littérature et se veut ainsi être une contribution à la recherche sera, dans un premier temps, présentée. Dans un second temps, les classifications individuelles seront traitées afin de déterminer le choix le plus probable et ainsi sélectionner le caractère représenté dans l'image.

2.3.1 Classification individuelle des caractères

Tel quel discuté lors du chapitre 1, la littérature tend présentement à utiliser les réseaux de neurones pour réaliser l'extraction des caractéristiques ainsi que la classification de l'image de manière automatique. Or, comme vu lors de la section 2.2 sur l'extraction des caractéristiques, cette recherche tente aussi d'utiliser la DCT, la transformée de Hough et la DWT pour orienter la caractérisation et tenter d'améliorer la classification. C'est pourquoi la classification est réalisée avec quatre réseaux de neurones, incluant l'image brute, par caractère comme l'on peut voir à la figure 28 pour la reconnaissance des caractères des chiffres. L'architecture pour les chiffres est d'abord explicitée, suivie par les lettres.

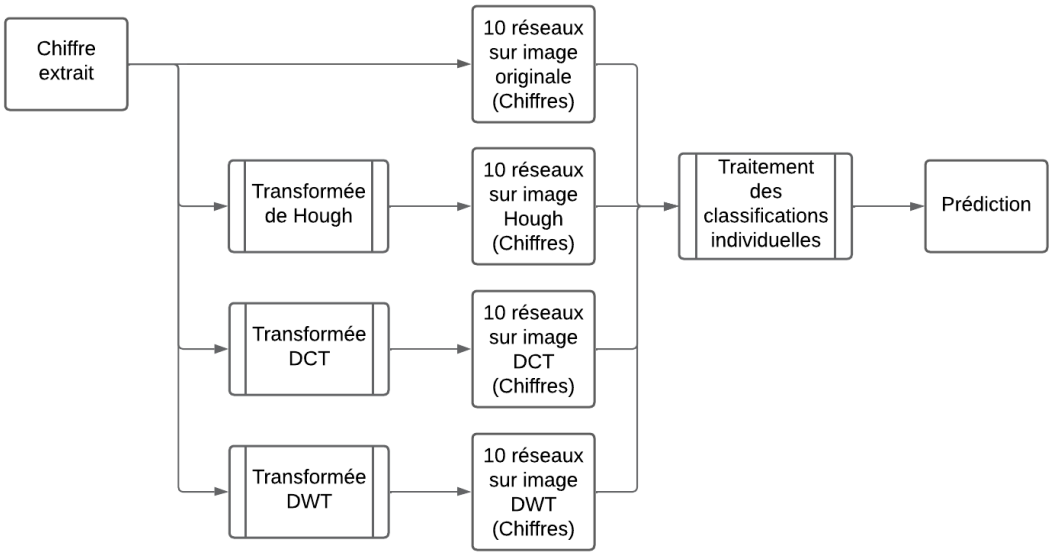

Figure 28. Extraction des caractéristiques et classification d'un chiffre selon la méthode proposée. Une fois l'image brute du chiffre extraite, les transformées Hough, DCT et DWT sont calculées selon les méthodes présentées précédemment et chacune de ces images sont présentées aux dix réseaux de neurones appropriés. Un dernier réseau de neurones reçoit ensuite les poids de chacun des réseaux précédents et établit une prédiction finale.

Cette méthode représente une seconde contribution de cette recherche à la littérature. En effet, au lieu d'utiliser un seul réseau de neurones convolutif de grandes dimensions pour réaliser l'extraction des caractéristiques et la classification, quatre réseaux de neurones par caractère sont utilisés dans une approche de type « Un contre tous ». Ainsi, seulement pour les chiffres, c'est un total de 40 réseaux de neurones entraînés individuellement qui sont utilisés (quatre réseaux pour chacun des dix chiffres). Cette approche est celle qui a permis d'obtenir le meilleur taux de reconnaissance lors d'expériences accessibles dans le répertoire de projet pour les semaines du 2022-11-07 au 2023-12-04, tout en permettant d'avoir une architecture parallèle permettant le traitement asynchrone visant à réduire l'attente entre chacune des étapes et accélérer le système. Cette approche permet donc théoriquement d'améliorer deux des objectifs du projet, soit le taux d'exactitude et la vitesse de traitement au coût d'un temps d'apprentissage plus long dû au grand nombre des réseaux. Une représentation de chacun des réseaux individuels se trouve à la figure 29.

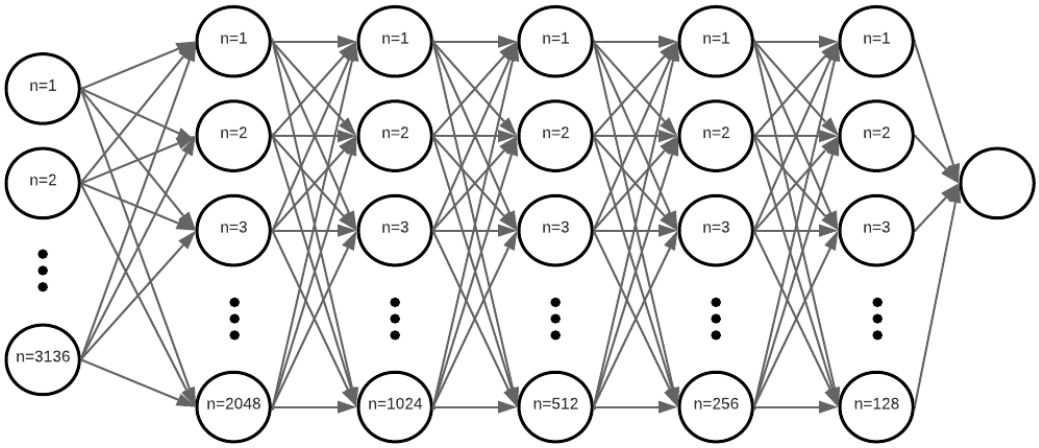

Figure 29. Architecture du réseau de neurones proposé pour chacun des caractères. 3136 neurones sont utilisés en entrée pour recevoir chacun des pixels de l'image brute (56x56). Cinq couches de neurones cachées sont ensuite utilisées (de 2048 à 128 neurones par couche en divisant par deux à chacune des couches)

Un réseau de neurones à connectivité totale entre les couches est utilisé pour chacun des caractères. Contrairement à la littérature favorisant plutôt les réseaux de neurones convolutifs (CNN), des réseaux de neurones à connectivité totale ont été choisis afin de ne poser aucune hypothèse sur le lien entre la position des pixels comme le fait un CNN compte tenu du fait que cette hypothèse n'est pas nécessairement valide pour la DCT et la DWT contenant du contenu fréquentiel. De plus, d'un point de vue théorique, le réseau de neurones à connectivité totale donnera un résultat équivalent à un CNN dans le scénario où l'approche minimise bel et bien l'erreur. Un autre intérêt du CNN est sa capacité à réduire la complexité de calculs d'une image. Cependant, comme l'on manipule ici des images de faible dimension (56x56) ce gain semble moins justifié, c'est pourquoi les réseaux de neurones à connectivité totale sont ici considérés. Finalement, concernant le nombre de neurones, 3136 neurones sont utilisés à l'entrée pour recevoir les images de dimensions (56x56), l'image est ainsi ramenée à une forme de vecteur colonne pour le présenter au réseau de neurones. Par la suite, des divisions par puissance de 2 sont réalisées afin de réduire progressivement le nombre de paramètres et la complexité de calculs jusqu'à la sortie pour chacune des cinq couches cachées. La réduction du nombre de neurones a aussi pour objectif de réduire les risques de

surapprentissage qui pourraient influencer négativement les résultats (Bishop, 2006). Au tableau 3 se trouve un résumé des hyperparamètres pour chacun de ces réseaux. Cette architecture est celle qui a, encore une fois, permis d'obtenir les meilleurs résultats expérimentaux en termes de taux de reconnaissance et les résultats d'entraînement se trouvent dans le répertoire d'expériences à la semaine du 2023-09-04. Concernant les neurones se trouvant dans les couches cachées du réseau de neurones, la fonction d'activation Tanh a été utilisée et son équation se trouve à l'équation 6. La fonction sigmoïde a, quant à elle, été utilisée comme fonction d'activation pour le neurone de sortie et son équation se trouve à l'équation 7. La principale distinction entre ces deux équations est que la fonction Tanh est bornée de -1 à 1, permettant ainsi aux neurones de favoriser ou défavoriser certains résultats. La fonction sigmoïde est, quant à elle, bornée de 0 à 1 ce qui permet une sortie normalisée. Dans le cas où un neurone de sortie est à zéro, il est alors peu probable d'avoir reconnu le caractère et à l'inverse, lorsque la sortie est près de 1, il est probable que le caractère a été reconnu. Pour la fonction sigmoïde, la valeur 0,5 représente une zone incertaine au niveau de la classification (Kinsley & Kukieła, 2020). D'autres fonctions d'activation pour les couches cachées ont été envisagées lors d'expérimentations telles que la fonction linéaire, le redresseur et ses alternatives, mais la fonction Tanh et son comportement non-linéaire obtenait de meilleurs résultats lors de ces essais. Une schématisation d'un neurone de sortie se trouve à la figure 30.

Équation 6 : Équation générale de la fonction d'activation Tanh bornée entre -1 et 1

$$Tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Équation 7 : Équation générale de la fonction d'activation sigmoïde (Tensorflow, 2023)

$$\phi(z) = \frac{1}{1 + e^{-z}}$$

Tableau 3. Résumé des hyperparamètres pour l'architecture de réseaux individuels proposée

Fonction d'activation (couches cachées)	Tanh [-1, 1]
Fonction d'activation (sortie)	Sigmoïde [0, 1]
Optimisateur	Adam
Nb. Epoch	30 (100 pour DCT)
Régularisateur (L2)	1,00E-05
Taux d'abandon	0,05
Taux d'apprentissage	1,00E-05
Taille de l'échantillon	32

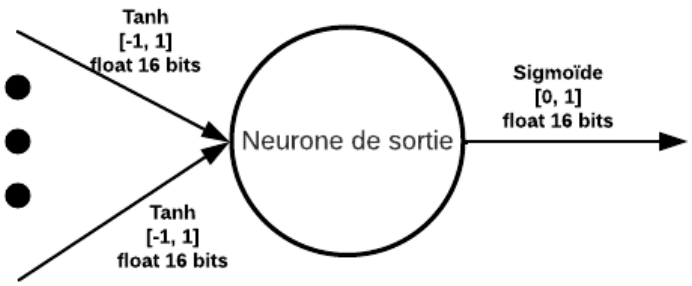

Figure 30 : Schématisation d'un neurone de sortie. Le neurone reçoit ainsi un nombre N d'entrées de -1 à 1 selon la fonction Tanh et permet d'obtenir un résultat entre 0 et 1. Les nombres sont tous exprimés en format 16 bits flottant afin de profiter des accélérations matérielles de la plateforme *CUDA*.

En abordant maintenant le cas des lettres, conformément à l'article de Lukasik (2021), de meilleurs taux de reconnaissance ont été obtenus en regroupant certaines lettres plus difficiles à distinguer. Expérimentalement, les lettres « i » et « l » ainsi que « g » et « q » sont des combinaisons qui étaient difficiles à distinguer et entraînaient un grand taux d'erreurs selon les résultats obtenus lors de la semaine du 2023-05-01. Ainsi, dans le cadre de la

solution proposée, il a été décidé de ne pas réaliser la distinction et d'utiliser un seul réseau de neurones pour reconnaitre une de ces combinaisons. Dans le même ordre d'idées de réduction de classes pour améliorer les résultats, de meilleurs résultats expérimentaux ont été obtenus en ne réalisant pas la distinction de la case sur certaines lettres dû à leurs formes semblables (par exemple la lettre « O »). Ces résultats sont disponibles dans le répertoire d'expérimentation pour la semaine du 2023-02-27 (sans distinction) et du 2023-03-20 (avec distinction) sur les lettres spécifiées. Un tableau résumant les lettres ayant une distinction majuscule/minuscule et celles n'ayant pas cette distinction se trouve au tableau 4. Ainsi, les lettres n'ayant pas de distinction à la case ont un seul réseau de neurones pour reconnaitre ce caractère toujours avec l'approche « Un contre tous », alors que les lettres avec distinctions de la case ont un réseau pour la lettre minuscule et un pour la lettre majuscule toujours selon la même approche et la même architecture.

Tableau 4. Lettres ayant une distinction à la case

Lettres sensibles à la case	A, B, D, E, F, G et Q, H, I et L, J, N, R, T
Lettres sans distinction de la case	C, K, M, O, P, S, U, V, W, X, Y, Z

De cette manière, on obtient, pour les lettres, 36 réseaux par type d'images, et donc, 144 réseaux différents en parallèle pour considérer l'ensemble des caractéristiques des lettres comme on peut voir à la figure 31.

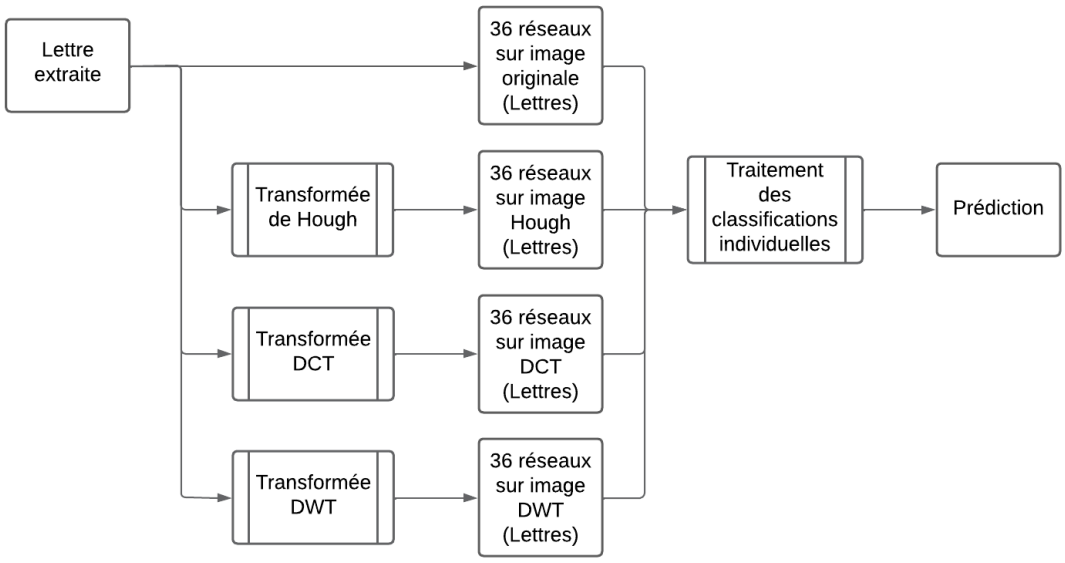

Figure 31 : Extraction des caractéristiques et classification d'une lettre selon la méthode proposée

Cependant, comme mentionné dans l'article de Lukasik (2021), plus le nombre de classes est élevé et moins les performances sont élevées. Par conséquent, il est préférable de réduire le nombre de classes autant que possible. Dans les cas où l'on connait le contenu attendu dans une case, on sait préalablement qu'on ne retrouvera pas de chiffres (ex. : nom) ou de lettres (ex. : numéro de téléphone) ou qu'il s'agit d'une combinaison précise des deux comme l'alternance dans le code postal. Ainsi, il n'est pas nécessaire de considérer les 184 réseaux lorsque l'on cherche à classifier un caractère. Dans le cas d'un numéro de téléphone, on peut ainsi utiliser seulement les 40 réseaux développés pour les chiffres, réduisant ainsi significativement le nombre de possibilités. Cette flexibilité à ne pas évaluer certaines classes est un autre des attraits de la méthode « Un contre tous » ici utilisée. Toutefois, dans les cas de figure où il n'est pas possible de réduire le nombre de réseaux selon le contexte comme avec l'adresse qui est une combinaison de lettres et de chiffres, il a été possible de remarquer expérimentalement que le système a tendance à confondre les chiffres et les lettres qui se ressemblent (ex. : « 1 » et « l ») comme on peut voir dans le dossier d'expérimentations à la semaine du 2023-10-30.

Afin de réduire ce problème, quatre nouveaux réseaux sont introduits pour distinguer les chiffres et les lettres (un pour chaque représentation du caractère soit l'image brute, la DCT, la DWT et la transformée de Hough). Ceux-ci sont appliqués avant les réseaux pour les chiffres et pour les lettres présentées précédemment afin de réaliser une pré-classification et orienter la détection permettant ainsi de réduire le nombre de classes à traiter. Ces quatre nouveaux réseaux ont été entraînés dans le but de classer le caractère comme une lettre ou un chiffre afin de ne traiter que le cas concerné. L'architecture du réseau est présentée à la figure 32 ainsi que les hyperparamètres d'apprentissage au tableau 5. Deux principales différences sont notables au tableau 5 en comparaison à ce qui a été présenté précédemment. Dans un premier temps, la fonction d'activation de sortie a été remplacée par la fonction *Softmax* et son équation se trouve à l'équation 8. Cette fonction permet, dans le cas d'une couche de sortie à plusieurs neurones, d'obtenir un niveau de confiance pour chacune des classes pour laquelle l'addition est normalisée. Ainsi, la somme des poids des sorties permet d'obtenir 1 et chacun des poids peut être visualisé comme la probabilité d'avoir un chiffre ou une lettre. La seconde différence se trouve au niveau de la taille de l'échantillon qui a été augmentée lors de l'entraînement. Ce paramètre a été augmenté principalement afin de réduire le temps d'entraînement.

Équation 8 : Équation de la fonction *Softmax*. La valeur attribuée à chacune des sorties correspond à l'exponentiel de la valeur du neurone divisé par la sommation des exponentielles des autres neurones sur la même couche (Tensorflow, 2023).

$$s(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

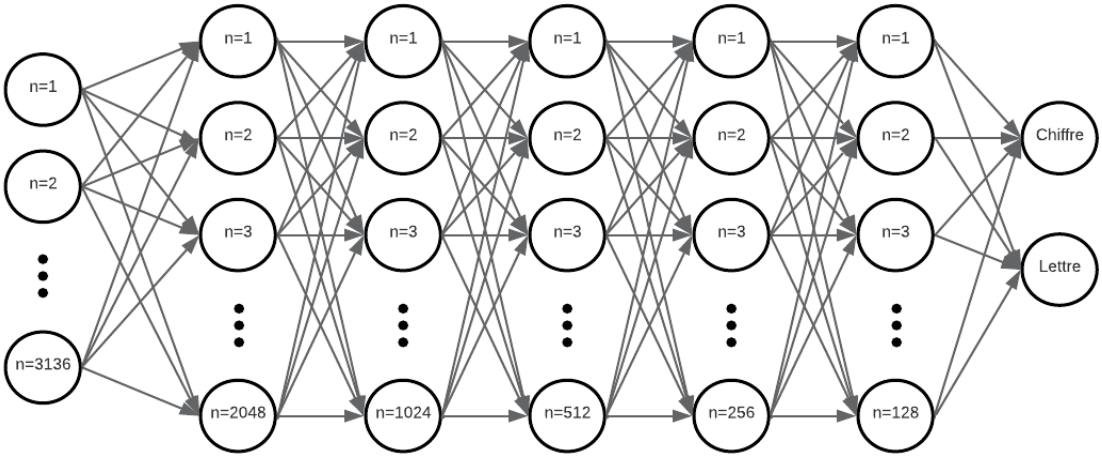

Figure 32 : Réseau de neurones proposé pour la distinction entre chiffres et lettres. Ce réseau est répété quatre fois pour chacune des représentations de l'image (brute, DCT, DWT et Hough).

Tableau 5 : Résumé des hyperparamètres pour l'architecture de réseaux réalisant la distinction entre chiffres et lettres

Fonction d'activation (couches cachées)	Tanh
Fonction d'activation (sortie)	Softmax
Optimisateur	Adam
Nb. Epoch	30 (100 pour DCT)
Régularisateur (L2)	1,00E-05
Taux d'abandon	0,05
Taux d'apprentissage	1,00E-05
Taille de l'échantillon	256

2.3.2 Traitement des classifications individuelles

À partir des sorties de chacun des réseaux individuels, il est alors possible de réaliser le traitement afin de déterminer quel caractère a été reconnu. L'un des défauts de l'approche « Un contre tous » est la perte d'informations entre les classes. En effet, comme chacune des classes est entraînée de manière individuelle, le réseau ne peut utiliser de relations entre les classes elles-mêmes qui l'aideraient dans sa classification. Dans l'objectif de réduire cette problématique, un réseau de neurones est utilisé à la sortie des multiples réseaux afin de considérer la sortie de chacun dans la sélection tel qu'illustré précédemment à la figure 28 ainsi qu'à la figure 31 avec le bloc « Traitement des classifications individuelles ». Cette architecture, de dimension bien inférieure à l'architecture présentée dans la classification individuelle, est illustrée à la figure 33 et est utilisée à deux reprises, une fois pour les chiffres et l'autre pour les lettres. La seule différence entre les deux réseaux entraînés se trouve au nombre de neurones d'entrées qui dépend du nombre total de réseaux pour chacun (40 pour les chiffres et 144 pour les lettres) ainsi que de la sortie qui dépend du nombre de classes dans chacun des cas (10 chiffres ou 26 lettres).

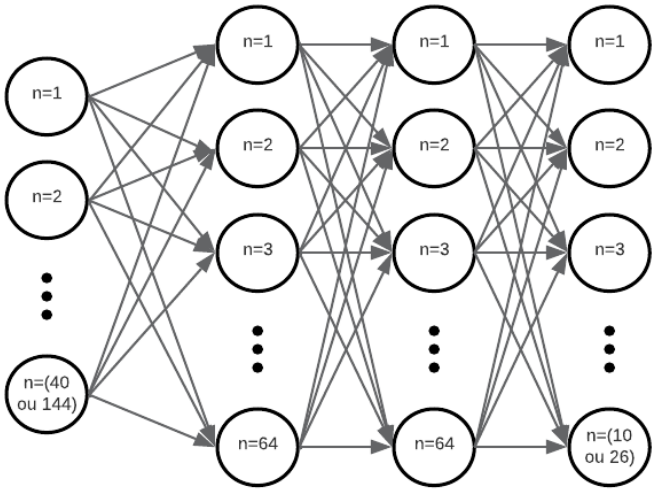

Figure 33. Réseau de neurones proposé pour le traitement des classifications individuelles Le nombre de neurones d'entrées dépend du nombre de classes total et le nombre de neurones de sorties correspond au nombre de chiffres ou de lettres.

De manière équivalente à ce qui a été vu précédemment, l'architecture de la figure 33 a été obtenue à la suite de différents essais expérimentaux de manière à déterminer les paramètres maximisant le taux de reconnaissance. Ces différents essais expérimentaux (7) se trouvent dans le dossier d'expérimentations du projet à la Semaine du 2023-09-11. Les hyperparamètres retenus se trouvent, quant à eux, au tableau 6.

Tableau 6. Résumé des hyperparamètres pour l'architecture de réseau servant au traitement des classifications individuelles

Fonction d'activation (couches cachées)	Tanh
Fonction d'activation (sortie)	Softmax
Optimisateur	Adam
Nb. Epoch	10
Régularisateur (L2)	1,00E-05
Taux d'abandon	0,50
Taux d'apprentissage	0,01
Taille de l'échantillon	256

Pour résumer l'ensemble du système de classification, le schéma à la figure 34 illustre une vision globale.

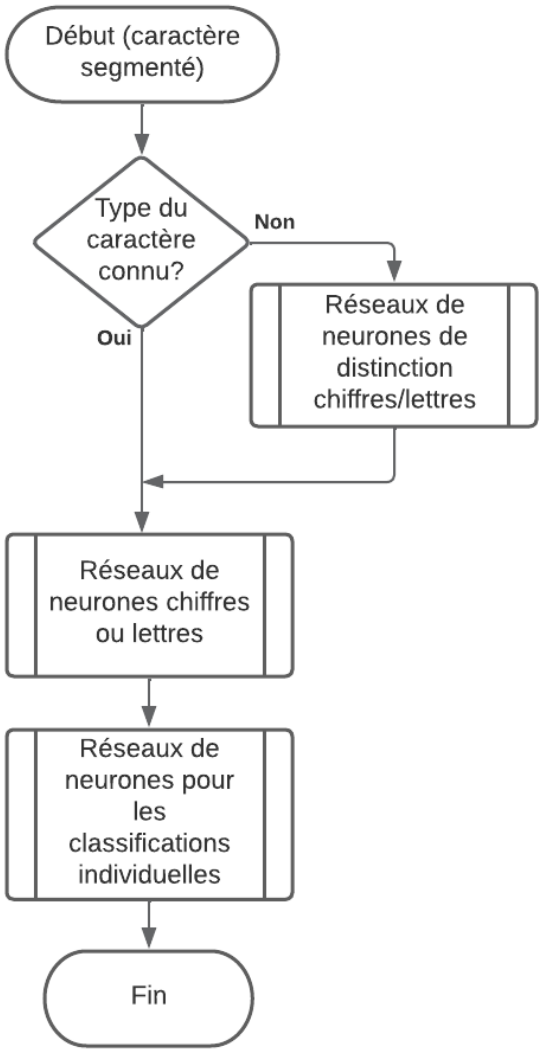

Figure 34 : Schématisation des différentes étapes constituant la classification pour un formulaire structuré. Par « caractère segmenté », on signifie ici des sous-images binaires de caractères idéalement isolés et centrés dans l'image. Les blocs « Réseaux de neurones » comprennent ici les parties d'extractions des caractéristiques (3) en plus de l'image brute utilisée.

2.3.3 Base de données pour entraînement

Afin de réaliser l'entraînement des réseaux de neurones présentés à la sous-section précédente, la base de données EMNIST a été utilisée (Cohen, Afshar, Tapson, & van Schaik,

2017). Cette base de données reprenant un format et une structure correspondant à la base de données MNIST (base de données de chiffres parmi les plus utilisées) contient en plus plusieurs centaines de milliers d'exemples de lettres manuscrites nécessaires à la réalisation du système désiré (Memon, Sami, Khan, & Uddin, 2020). Ainsi, celle-ci est bien adaptée aux besoins du projet et contient un total de 814 255 caractères séparés en 62 classes non balancées permettant ainsi d'épargner beaucoup de temps à extraire et étiqueter des données manuellement pour un apprentissage supervisé.

Pour les réseaux de neurones permettant de classifier les chiffres, la division « Digits » est utilisée et contient 280 000 images de chiffres séparés en 10 classes (0 à 9). Concernant les réseaux de neurones pour les lettres sans distinctions de la case, c'est la division « Letters » qui a été utilisée avec ses 145 600 images de lettres divisées en 26 classes balancées. Finalement, dans le cas des réseaux de neurones pour lettres avec une distinction à la case, c'est la division « ByClass » qui a été utilisée contenant contenant 411 302 images de lettres divisées en 52 classes non-balancés. Pour les expérimentations des semaines du 2022-11-07 jusqu'au 2023-06-12 inclusivement dans le répertoire d'expérimentation, les essais ont été réalisés directement avec les divisions de ces bases de données qui permettait d'obtenir des résultats satisfaisants selon l'état du projet à chacune de ces semaines.

Cependant, lors de la semaine du 2013-06-12, en observant manuellement les données dans le but d'analyser les erreurs et cibler des améliorations à apporter, des erreurs d'étiquetage ou encore des caractères non identifiables ont été repérés. Ces erreurs, en proportion suffisamment faible pour ne pas être repérées lors de coup d'œil rapide dans la base de données ou dans les résultats préliminaires représentent cependant une problématique pour la validité des résultats obtenus jusqu'alors. En effet, l'utilisation d'une base de données contenant des erreurs lors d'un apprentissage supervisé amène un risque d'introduire des faussetés pouvant avoir un effet non négligeable sur l'apprentissage et la capacité de généralisation des réseaux de neurones biaisant les résultats finaux. Ainsi, pour s'assurer la validité des résultats pour la suite du projet, il a été nécessaire d'analyser plus en profondeur

la base de données afin de réduire les risques associés. Un résumé des divisions utilisées de la base de données EMNIST est présenté au tableau 7.

Tableau 7 : Récapitulatif des divisions pour la base de données EMNIST. Les images sont de taille 28 par 28 pixels en nuance de gris et encodés sur 8 bits non signés

Division	Nombre de classes	Images par classe	Nombre total d'images
Digit Dataset	10 (0 à 9)	30 000 (balancé)	300 000
Letters Dataset	26 (A à Z)	6 000 (balancé)	156 000
By_Class Dataset	52 (A à Z et a à z)	Variable (non balancé)	411 302 (excluant les chiffres)

Or, la grande quantité d'images rend une vérification manuelle beaucoup trop longue. Ainsi, pour filtrer les différentes sources d'erreurs de manière semi-automatique, un réseau de neurones a été utilisé de nouveau. L'architecture du réseau ayant été utilisée se trouve à la figure 35 et les hyperparamètres se trouvent au tableau 8. Il est toutefois important de noter que cette architecture et les hyperparamètres n'ont ici pas été optimisés dû à la difficulté d'évaluer les performances à détecter les erreurs sans avoir à revérifier les résultats manuellement. L'objectif ici était surtout de détecter rapidement et facilement de possibles erreurs, mais une recherche plus poussée aurait pu être réalisée. La seule différence entre le réseau de filtrage pour les chiffres et celui pour les lettres est le nombre de neurones à la sortie représentant le nombre de classes possibles (10 chiffres, 26 lettres sans distinction majuscule/minuscule et 52 lettres avec la distinction majuscule/minuscule).

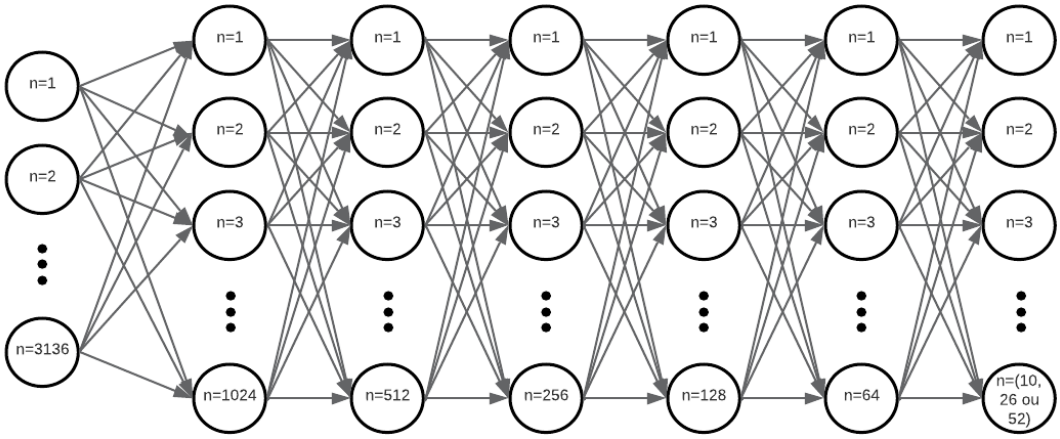

Figure 35. Architecture de réseau de neurones pour le filtrage de la base de données. La première couche cachée contient 1024 neurones et chaque couche cachée diminue par la suite d'un facteur 2. La couche de sortie contient un nombre de neurones égal au nombre de classes

Tableau 8. Résumé des hyperparamètres pour l'architecture de réseau servant au filtrage de la base de données

Fonction d'activation (couches cachées)	Tanh
Fonction d'activation (sortie)	Softmax
Optimisateur	Adam
Nb. Epoch	15
Régularisateur (L2)	1,00E-05
Taux d'abandon	0,50
Taux d'apprentissage	1,00E-04
Taille de l'échantillon	128

Pour réaliser le filtrage, 50% de la base de données sélectionnée de manière aléatoire a été utilisé pour faire l'apprentissage du réseau. Une fois l'entraînement terminé, le 50% de la base de données restant a été utilisé afin d'obtenir les prédictions pour chacune des images en les présentant au réseau. De ces images, les erreurs de détection ont été retenues et

revérifiées manuellement. Les erreurs d'étiquetage ont ainsi été corrigées et les images mal tracées ont été supprimées après une vérification manuelle. Plus précisément, seulement les cas impossibles à identifier par un observateur ont été supprimés afin d'éviter d'ajouter un biais en modifiant la base de données en retirant des cas plus difficiles ou tout simplement moins bien tracés. De plus, pour réduire les risques d'altérer arbitrairement des images qui fausseraient la suite des résultats, chacune des images ayant été réidentifiées ou retirées ont été notées avec un commentaire mentionnant la correction apportée. De cette manière, une traçabilité a été conservée et est accessible dans le répertoire d'expérimentation pour les semaines du 2023-06-19 et du 2023-06-26 pour chacune des itérations.

Une fois la vérification manuelle réalisée, le réseau de neurones a été réinitialisé et un second apprentissage a été fait sur le 50% de la base de données ayant été utilisé pour les prédictions moins les images supprimées. Similairement, le second apprentissage a aussi permis de faire des prédictions sur le premier 50% qui avait servi au premier apprentissage et les erreurs de détection ont été revérifiées encore une fois manuellement pour corriger les erreurs d'étiquetage et les chiffres illisibles. Cette même opération a été répétée à trois reprises sur chacune des moitiés. Au-delà de trois répétitions, il a été jugé que le temps investi pour le nombre d'erreurs ne justifiait plus l'opération puisque le nombre d'erreurs diminuait rapidement entre chaque itération. Grâce à cette méthode, pour la base de données sur les chiffres, 215 erreurs (environ 0,08%) ont été détectées et corrigées réduisant la base de données à 279 785 images.

De manière exactement équivalente, la division « Letters » a aussi été filtrée en suivant la même méthodologie que pour les chiffres. Seulement les cas auxquels il n'était pas possible d'identifier clairement la lettre et qui étaient hors de tout doute des erreurs ont été retirés ou renommés. La traçabilité des modifications apportées se trouve quant à elle dans le répertoire d'expérimentation pour la semaine du 2023-07-25 et du 2023-07-31. Ainsi, c'est 1064 erreurs (environ 0,73%) qui ont pu être corrigées amenant le total d'images restantes à 144 806.

Similairement, la division « ByClass » utilisée pour les lettres avec une distinction de la case a été filtrée. Toujours en suivant la même méthodologie et en n'altérant que les erreurs certaines, mais en ne faisant qu'une seule itération par faute de temps, 2105 erreurs (environ 0,51%) ont été corrigées amenant le total à 410 468 images disponibles pour l'entraînement. Comme pour les chiffres ainsi que les lettres, les modifications apportées ont été compilées afin d'assurer la traçabilité dans le répertoire d'expérimentation à la semaine du 2023-08-07. Afin d'illustrer les erreurs d'étiquetage ainsi que les images « illisibles », les figures 36 et 37 présentent certains exemples typiques tirés des dossiers d'expérimentations réalisés pour les semaines allant du 2023-06-20 au 2023-08-07. À la figure 36, il est possible d'observer quatre caractères dont dans l'ordre un « 2 » étiqueté en « 3 », un « 7 » étiqueté en « 8 », un « e » étiqueté en « I » et un « a » étiqueté en « A ». Concernant les caractères illisibles, à la figure 37 sont présentés quatre caractères de gauche à droite, selon la base de données, un « 8 », un « 4 », un « x » ainsi qu'un « u ». Ces caractères sont difficiles voire impossible à reconnaitre même pour un observateur humain.

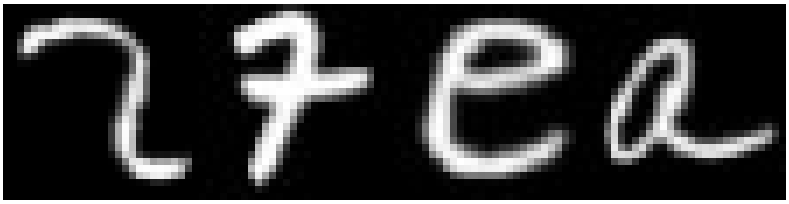

Figure 36. Exemples typiques d'erreurs d'étiquetage renommées dans la base de données. Selon la base de données initiale, il s'agirait dans l'ordre des caractères : 3, 8, I et A.

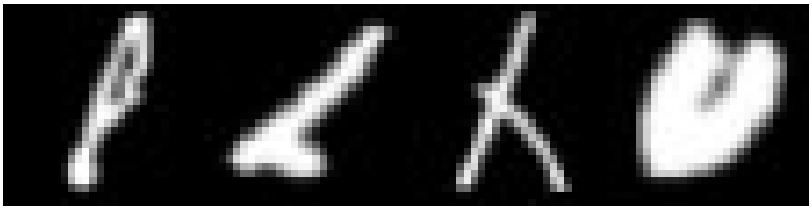

Figure 37. Exemple de caractères jugés « illisibles » retirés de la base de données étiquetés comme étant dans l'ordre les caractères : 8, 4, X et u.

Une fois les trois divisions de la base de données utilisées pour le projet filtrées, chacune d'elles a été séparée aléatoirement en trois groupes pour être le plus objectif possible dans la phase d'entraînement. Ainsi, 80% de la division filtrée sert à la phase d'entraînement des réseaux de neurones. Un autre 10% de validation, sert quant à lui à évaluer la base de données durant la phase d'apprentissage pour éviter des problèmes de surapprentissage et évaluer l'évolution des performances entre chaque epoch. Finalement, un dernier 10% des images sert à titre de référence afin de comparer les résultats entre chacun des essais expérimentaux. Ce dernier 10% n'ayant jamais été présenté au préalable autant aux réseaux individuels qu'au réseau traitant les classifications individuelles à la section 2.3.2, celui-ci permet d'avoir une référence plus objective et indépendante des résultats.

Malheureusement, le filtrage de la base de données ainsi que la nouvelle séparation fait que le système réalisé dans le cadre de ce projet de recherche ne peut être comparé directement avec la littérature dû à la référence qui est maintenant différente. Cependant, l'ordre de grandeur devrait théoriquement demeurer comparable puisque le nombre d'erreurs corrigées sur chacune des divisions est inférieur à 1%.

2.4 POST-TRAITEMENT

Les différentes étapes de détection et de correction des erreurs seront présentées lors de cette sous-section. Cette étape est nécessaire afin de réduire le risque d'erreurs et le nombre de corrections nécessaires par un opérateur externe. De plus, les détails en lien avec une nouvelle phase d'apprentissage intégrée au post-traitement permettant au système de s'améliorer progressivement même au-delà du présent projet de recherche sont aussi présentés.

2.4.1 Détection et correction des erreurs

Concernant la détection et la correction des erreurs, trois méthodes de vérification ont été implantées pour le projet, mais plusieurs autres pourraient être ajoutées dans un volet futur selon les besoins en s'inspirant des résultats de l'article *Deep statistical analysis of OCR errors for effective post-OCR processing* (Nguyen, Jatowt, Coustaty, Nguyen, & Doucet, 2019). La première méthode consiste tout simplement à comparer le mot détecté avec un dictionnaire contenant une liste des valeurs possibles ainsi que la fréquence à laquelle ce mot est utilisé. Pour ce faire, lorsque tous les caractères dans une case ont été reconnus lors de la classification, ceux-ci sont regroupés selon leur ordre d'apparition. Le mot ainsi constitué est ainsi comparé avec chacun des mots se trouvant dans le dictionnaire en calculant la distance de Levenshtein. Le mot ayant la distance de Levenshtein la plus faible est ainsi considéré comme étant le mot réel se trouvant dans la case. En cas d'égalité, la fréquence d'apparition est utilisée afin de sélectionner le mot le plus probable. Ce type de correction est principalement appliqué aux pays, provinces, aux villes, aux noms et aux prénoms, mais est surtout efficace lorsque le dictionnaire contient un registre plus limité comme les pays et les provinces.

Le second type de détection et correction est spécifique aux dates. Dans le cas où une date est demandée sur le formulaire, on vérifie que la date détectée correspond à une date existante plausible. On rejette ainsi les scénarios où l'on aurait un 13^{e} mois ou encore une date de naissance qui serait incohérente. Cependant, ces différentes erreurs sont difficiles à corriger de manière automatique par le système. C'est pourquoi, pour ce type d'erreur détecté, un message est présenté à l'opérateur afin que celui-ci puisse vérifier le document et apporter la correction requise.

Le dernier type de détection et correction appliqué dans le cadre du système consiste principalement à détecter des anomalies dans les adresses en utilisant certaines particularités. Dans un premier temps, dans le cas de figure ou un chiffre serait détecté dans l'adresse, mais n'ayant que des lettres comme voisins, on peut supposer que le réseau cherchant à réaliser la

distinction entre les chiffres et les lettres a commis une erreur et que le caractère recherché est probablement une lettre puisqu'il ne s'agit pas d'un format commun au Canada. Il est alors possible de refaire la classification avec les réseaux pour les lettres et vérifier si le poids obtenu est au-delà d'un certain seuil. En plus de cette vérification, l'information dans la case « pays » est beaucoup plus restreinte en termes de possibilités, celle-ci est donc utilisée comme référence pour déterminer la liste possible de provinces/états/territoires que l'on peut ensuite utiliser comme dictionnaire (première méthode de détection et correction), puis répéter la même étape pour générer la liste possible des villes possibles comme dictionnaire. Cette étape a ainsi principalement pour but d'assurer une cohérence entre le pays, la province et la ville. À tout cela s'ajoute aussi une détection sur le code postal en vérifiant le format de celui-ci pour les adresses canadiennes (alternance lettres/chiffres et six caractères). Une fois le pays, la province/état/territoire, la ville et le code postal vérifiés, une contre-vérification supplémentaire pourra être ajoutée au système en comparant les informations avec une interface de programmation telle que « Address Validation API » proposé par l'entreprise *Google LLC*. Ce dernier volet n'a pas été implanté dans le cadre du projet compte tenu de l'évaluation des coûts et les démarches requises qui dépassent le mandat du projet de recherche, mais cette option permettrait d'ajouter un volet de vérification supplémentaire et réduire les erreurs si celles-ci s'avèrent trop fréquentes.

2.4.2 Nouvelle phase d'apprentissage

Dans l'objectif de permettre une amélioration du taux de reconnaissance à long terme et minimiser les erreurs provenant des réseaux de neurones, plusieurs phases d'apprentissage peuvent être envisagées de nouveau sur une période plus ou moins longue. En effet, comme mentionné dans l'introduction générale, c'est plus de 30 millions d'images de documents qui sont numérisées par année par la *DIORG*. Or, comme vu dans l'œuvre de Goodfellow (2016) ou dans l'œuvre de Bishop (2006), la quantité de données a une importance prépondérante dans l'apprentissage à l'aide de réseaux de neurones. En général, plus le nombre de données

d’apprentissage augmente et plus le réseau tend à bien généraliser l’information et ainsi améliorer sa classification. À partir de ce raisonnement, l’une des manières de réduire l’erreur du système d’OCR est ainsi de profiter du grand volume de données traitées pour augmenter le nombre de données d’entraînement des réseaux de neurones. Pour ce faire, une fraction des données classifiées, une fois validée par un opérateur humain afin d’éviter d’introduire des erreurs, pourront être introduites dans notre base de données selon le type de caractères et servir à une nouvelle phase d’entraînement après une période prédéfinie. Les nouveaux réseaux de neurones entraînés hors des périodes d’opérations peuvent ensuite remplacer les anciens et démarrer une nouvelle période tel. Grâce à ce cycle, on augmente ainsi potentiellement de plusieurs millions d’images la taille de la base de données ce qui devrait, théoriquement, améliorer l’étape de classification et réduire les erreurs du système en améliorant la généralisation et en s’adaptant aux données réelles. Une schématisation est présentée à la figure 38.

Cependant, cette étape ne pourra être évaluée dans le cadre du projet dû à son effet à long terme et le temps nécessaire à acquérir un nombre adéquat de caractères. Cette option est ainsi présentée ici, mais pourra être appliquée dans le futur lorsque suffisamment de données auront été obtenues. La période est aussi un paramètre à adapter si l’on souhaite maximiser les performances. Cependant, comme la reconnaissance et la validation peuvent se faire en parallèle à un cycle d’apprentissage sans créer de période d’arrêt (par exemple sur une autre machine), cette approche pour réduire le nombre d’erreurs ne devrait pas affecter la vitesse de traitement, ce qui est en cohérence avec les objectifs établis.

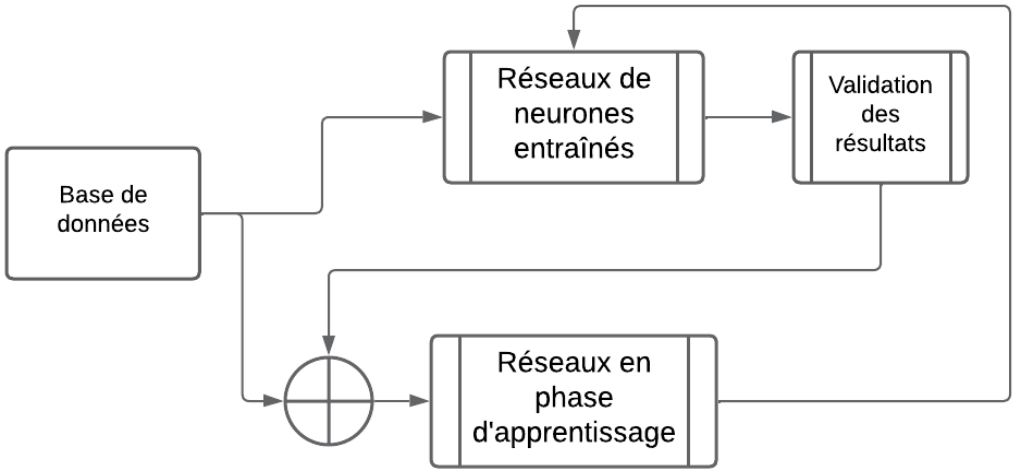

Figure 38. Système proposé à apprentissage répété. Les nouvelles données validées par un opérateur sont réintroduites dans la base de données et une nouvelle phase d'entrainement est répétée à une certaine fréquence, permettant ainsi au réseau d'apprendre sur des données réelles.

2.5 CONCLUSION

Pour conclure, le présent chapitre a permis de définir et d'expliciter les différentes solutions qui ont été retenues pour le système de reconnaissance de caractères sur des documents structurés. Dans le cas du prétraitement, l'algorithme *ORB* permet de corriger l'alignement et l'inclinaison dans les formulaires en le comparant à un modèle. Un fois le formulaire aligné et dilaté, une soustraction est réalisée entre le formulaire et le modèle et le résultat est filtré permettant ainsi d'identifier les informations inscrites et d'extraire chacune des cases.

Au niveau de la segmentation, un seuil adaptatif local est d'abord utilisé comme première approximation afin de vérifier si la case est vide ou non. Dans le cas où la case n'est pas vide, la méthode Otsu est utilisée pour amener l'image en format binaire. Une série de 15 à 20 dilatations est réalisée afin de détecter plus précisément la position du texte dans la case en cherchant le plus grand contour. La région du plus grand contour de la case est par la

suite extraite et une nouvelle détection de contours est réalisée afin de segmenter les caractères.

Au niveau de l'extraction des caractéristiques, la DCT, la DWT et la transformée de Hough sont utilisés pour caractériser l'image au niveau spatial (transformée de Hough), au niveau fréquentiel (DCT) ainsi que pour la combinaison des deux (DWT) ce qui permet ainsi d'avoir quatre représentations différentes pour caractériser l'image.

Au niveau de la classification des images, c'est un total de 190 réseaux de neurones différents (40 pour les chiffres, 144 pour les lettres avec et sans distinctions de cases et 4 pour distinguer les lettres et les chiffres) qui sont utilisés. Deux réseaux de neurones supplémentaires sont utilisés pour traiter chacune des classifications individuelles des réseaux de neurones (l'un pour les chiffres et l'autre pour les lettres). Au niveau du post-traitement, un dictionnaire, la restriction des cases ainsi que le contexte sont utilisés afin de réduire les possibles sources d'erreurs et proposer des corrections. Afin d'assurer une amélioration graduelle du système d'OCR, un système a apprentissage répété est suggéré.

Ainsi, pour les objectifs mentionnés dans l'introduction général, l'architecture proposée permet de réaliser une reconnaissance des caractères de manière partiellement asynchrone tout en réduisant les risques d'erreurs ce qui répond partiellement au premier objectif de recherche. De plus, la méthode utilisée ne nécessitant qu'un modèle du formulaire au préalable et les coordonnées des différentes cases lors du prétraitement permettent à des opérateurs non spécialisés d'ajouter de nouveaux formulaires facilement ce qui répond aux second et cinquième objectifs secondaires fixés lors de l'introduction générale. La capacité du système à détecter de possibles erreurs lors de la classification et du post-traitement et de le signaler permet de remplir le troisième objectif secondaire et partiellement quatrième auquel une interface doit encore être définie par le client à l'interne. La répétition continuelle de phases d'apprentissage à l'aide des nouvelles données à la section 2.4.2 permet, quant à elle, de répondre au sixième objectif secondaire. Par conséquent, le système d'OCR proposé permet de répondre à l'ensemble de l'objectif primaire et des objectifs secondaires visés. Une interface reste tout de même à définir (premier et quatrième objectifs secondaires), mais

celle-ci sera développée en interne afin d'assurer d'une meilleure homogénéité avec les systèmes actuels. Concernant le premier objectif primaire, les performances du système en termes de taux d'exactitude et de temps de traitement seront évaluées lors du chapitre suivant afin de déterminer l'atteinte complète de l'objectif.

CHAPITRE 3
PERFORMANCE DU SYSTÈME DE RECONNAISSANCE SUR DOCUMENTS STRUCTURÉS

Ce chapitre présente l'étude des performances sur un formulaire structuré du système de reconnaissance optique de caractère explicité au chapitre précédent. L'objectif est de quantifier le taux d'exactitude et le temps de traitement sur de l'écriture typographique ainsi que manuscrite afin de de déterminer si les objectifs énoncés dans l'introduction générale ont été atteints. Pour débuter, le matériel utilisé et le protocole d'expérimentation seront explicités afin de permettre une reproductibilité de l'expérience avec la programmation se trouvant aux annexes V à XIII et comparer avec les résultats individuels se trouvant aux annexes III et IV. Par la suite, les résultats moyens pour l'écriture typographique et manuscrite seront présentés succinctement pour ensuite valider l'atteinte ou non des objectifs selon les résultats obtenus et conclure sur différentes pistes de solutions à investiguer afin d'améliorer les performances dans de futures recherches.

3.1 MATERIEL ET PROTOCOLE D'EXPERIMENTATION

L'ensemble des expériences présentées lors de ce chapitre a été réalisé sur un ordinateur portable *LENOVO* avec un processeur *Intel Core i9-10885H*, une carte graphique *NVIDIA Quadro RTX 3000* et 32 Go de mémoire vive cadencé à 2933 MHz. D'un point de vu logiciel, l'environnement *Visual Studio Code* est utilisé avec les modules Python 3.10.0, *OpenCV* 4.7.0.72, *Tensorflow* 2.10.1 et *CUDA 11.4.0* lors des essais. Ces paramètres sont essentiels afin d'établir un référentiel pour évaluer les différents temps d'exécutions qui seront présentés dans la suite de ce chapitre et permettre une répétabilité de l'expérience. Concernant les images, celles-ci sont présentées aux réseaux de neurones en format binaire sous forme de liste en une seule dimension, les réseaux sont opérés avec une précision de 16

bits flottant et les résultats sont écrit dans un fichier CSV sur le disque dur de l'ordinateur. Cependant, les détails en lien avec la configuration et les manipulations réalisées par la carte graphiques sont inconnus comme ceux-ci sont gérés par la plateforme *CUDA*.

Au niveau de l'expérimentation, celle-ci se divise en deux volets distincts, soit la reconnaissance de caractères sur des formulaires avec une écriture typographique ainsi que ceux avec une écriture manuscrite. Afin de réduire la variance et permettre d'établir un comparatif des performances, un formulaire identique a été sélectionné pour les deux volets. Ce formulaire, provenant d'un projet réel actuellement traité à la *DIORG*, permet ainsi de simuler des conditions plus représentatives de la réalité. Comme il est possible de constater aux annexes III et IV, ce formulaire structuré a aussi comme intérêt d'avoir une bonne variété d'informations et de types de cases (texte uniquement, chiffres uniquement, case à cocher, combinaison lettres/chiffres et codes postaux) permettant d'étudier les performances sur l'ensemble de ces différents types à l'aide d'un formulaire unique.

Concernant les formulaires typographiques, les différents formulaires utilisés pour l'étude des performances se trouvent à l'annexe III avec les performances individuelles pour chacun présentés de manière successive. Pour chacun des quatre (4) formulaires (deux pour chaque langues officielles), des informations fictives ont été générées de manière aléatoire afin de simuler une demande et éviter de favoriser certains caractères ou données. Certaines cases ont parfois été laissées vides afin de vérifier si le système détecte bel et bien une case vide. Il convient de préciser qu'un formulaire contient rarement l'entièreté des cases remplies et que certaines cases sont indirectement déjà étudiées (ex : adresse postale, province et pays) par leur apparition répétée à différents endroits. Une fois les formulaires remplis à l'aide du logiciel *Power PDF Advanced*, ceux-ci ont été imprimés à une résolution de 300 points par pouce (DPI) en format couleur, puis numérisés à une résolution de 600 DPI en format TIFF sans compression. L'image numérisée a ensuite été directement utilisée par le système de reconnaissance optique de caractères sans modification et a permis d'obtenir les tableaux se trouvant aussi à l'annexe III comme fichiers de sortie. Pour chacun des tableaux, les lignes représentent une case dans le formulaire avec son contenu attendu et le type de caractères

attendus. Pour chacune des cases reconnues, une correction est suggérée en suivant les étapes de post-traitement détaillées précédemment lors du chapitre 2. L'exactitude avant et après post-traitement sont par la suite comparés en évaluant le nombre de caractères identifiés correctement en comparaison à la valeur réelle du formulaire. Concernant les temps mesurés, ceux-ci ont été obtenus à l'aide de la fonction *time()* de la bibliothèque *time* au début et à la fin des étapes mesurées. De plus, les erreurs de détections ont été identifiés en jaune afin de les mettre en évidence pour la section 3.2.

Une démarche similaire a été appliquée pour les formulaires manuscrits et ceux-ci se trouvent à l'annexe IV. La principale différence se trouve ici dans la méthode de remplissage des formulaires. Afin d'obtenir un échantillon plus varié d'écritures et de réponses, quatre (4) personnes sélectionnées aléatoirement ont rempli à la main trois (3) formulaires chacun. Les instructions qui leur ont été données sont les suivantes :

- N'utilisez que des informations fictives, mais plausibles;
- Remplissez un formulaire avec une main d'écriture appliquée;
- Remplissez un formulaire avec votre main d'écriture habituelle;
- Remplissez un formulaire avec une main d'écriture « rapide »;
- Variez vos réponses et vos manières de faire pour englober plusieurs cas;
- Ne communiquez pas entre vous afin de ne pas influencer les résultats.

De cette manière, douze (12) formulaires ont été recueillis avec des caractéristiques et des réponses différentes. À l'annexe IV, on remarque principalement que certains formulaires utilisaient différents outils tels que le stylo (noir, rouge et vert) et le plomb, que certains sont écrits en écriture cursive et d'autres en écriture scripte permettant ainsi de tester le système dans plusieurs cas différents. Un résumé de ces douze (12) cas est présenté au tableau 9 ci-dessous.

Tableau 9 : Résumé des caractéristiques des formulaires avec caractères manuscrits se trouvant à l'annexe IV. Les six premiers formulaires avec une écriture script ont été considérés dans l'évaluation des performances, alors que les six derniers ont été rejetés ne répondant pas à l'une des hypothèse générale ou représentant un cas extrême typiquement rejeté.

nº formulaire	Écriture (script vs cursif)	Type de crayon	Couleur du crayon	Considération dans l'évaluation des performances	Justification
1	Script	Stylo	Noir	Oui	
2	Script	Stylo	Noir	Oui	
3	Script	Stylo	Noir	Oui	
4	Script	Stylo	Noir	Oui	
5	Script	Stylo	Noir	Oui	
6	Script	Stylo	Noir	Oui	
7	Script	Stylo	Vert	Non	Non-respect d'une hypothèse générale (4e hypothèse)
8	Cursif	Stylo	Noir	Non	Non-respect d'une hypothèse générale (4e hypothèse)
9	Cursif	Stylo	Rouge	Non	Non-respect d'une hypothèse générale (4e hypothèse)
10	Cursif	Plomb	Gris à intensité variable	Non	Non-respect d'une hypothèse générale (4e hypothèse)
11	Script	Plomb	Gris à intensité variable	Non	Cas extrême (typiquement rejeté lors de la numérisation)
12	Script	Plomb	Gris à intensité variable	Non	Cas extrême (typiquement rejeté lors de la numérisation)

3.2 PERFORMANCES DU SYSTEME SUR FORMULAIRE AVEC CARACTERES TYPOGRAPHIQUE

3.2.1 Analyse des résultats

À partir des différents formulaires obtenus et leurs résultats associés se trouvant à l'annexe III, un tableau cumulatif contenant la moyenne des résultats pour chacune des cases est présenté au tableau 11 pour les quatre formulaires avec caractères typographiques. La police et la taille utilisées pour ces quatre formulaires sont celles par défaut du logiciel du logiciel *Power PDF Advanced* soit la police Myriad Pro en taille 10 points. Un exemple de cette police en taille 10 points est présenté au tableau 10. En jaune sont mises en évidences les cases avec une exactitude après post-traitement inférieure au minimum de 85% fixé comme objectif lors de l'introduction générale.

Tableau 10 : Comparaison des polices d'écriture Times New Roman utilisé pour rédiger ce mémoire avec la police Myriad Pro utilisé pour remplir les formulaires avec caractères typographiques pour différents caractères de base.

	Times New Roman (10 points)	Myriad Pro (10 points)
Chiffres	0123456789	0123456789
Majuscules	ABCDEFGHIJKLMNOPQRSTUVWXYZ	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Minuscules	abcdefghijklmnopqrstuvwxyz	abcdefghijklmnopqrstuvwxyz

À partir du tableau 11, il est possible d'observer que le taux moyen d'exactitude obtenu post-traitement pour de l'écriture typographique est de 91,99% et que les 25 cases ont nécessité un temps de traitement total de 132,55 s parmi lesquels 26,77 s ont été consacrées au post-traitement. Ce résultat global permet dans un premier temps de valider que le système obtient bien, en moyenne, un taux de reconnaissance supérieur à 85% comme spécifié dans les objectifs de l'introduction générale. Cependant, quatre (4) cases ont un taux d'exactitude plus faible que ce seuil fixé, soit le rang, l'appartement ou l'unité, l'adresse postale et le code postal. Pour comprendre les différentes erreurs rencontrées, il est possible d'observer individuellement les erreurs obtenues pour chacun des formulaires à l'annexe III. À partir de

ceux-ci, il est possible de voir que trois (3) des erreurs de détection présentes après post-traitement sur l'année de naissance, le code postal et l'adresse d'envoi pour le premier formulaire typographique aux pages 123 et 124 sont uniquement dues à une confusion entre le chiffre un (1) et le sept (7) ce qui amènerait seulement une erreur restante sur l'adresse d'envoi en supposant que ce problème ne serait pas présent. De manière équivalente sur le second formulaire aux pages 125 et 126, cette même confusion apparaît six (6) fois avec cinq (5) autres confusions supplémentaires entre le un (1) avec la lettre « l », la lettre « i » ou la lettre « t » ou encore le chiffre sept (7) avec le chiffre deux (2). Ces mêmes confusions se trouvent aussi sur le troisième et le quatrième formulaire aux pages 127 à 130 à différents niveaux ce qui représente la majorité des erreurs de détection dans le cas des formulaires typographiques. Ces erreurs surviennent principalement dans les quatre (4) cases mentionnées précédemment dû au fait que celles-ci ne limitent pas à un seul type de caractère (chiffres ou lettres) favorisant ainsi l'apparition de certaines de ces confusions dans le résultat final. Un exemple de ces confusions sont visibles à la figure 39.

Figure 39 : Exemple extrait du second formulaire structuré avec caractères typographiques à l'annexe III. Pour la valeur « 300 Wallinger Ave », le texte « 300 Wa111nger Ave » est reconnu montrant une confusion entre la lettre « l » et la lettre « i » avec le chiffre « 1 »

Au niveau du temps de traitement, il est possible de constater que le système prend en moyenne 4,02 s par case (132,55 s pour 25 cases) alors que l'objectif visé était de 4,00 s par case (une minute pour quinze (15) cases). Le système s'avère donc légèrement plus lent que la limite fixée dans la définition du projet. Sur ce temps total moyen, 20,19% est utilisé uniquement pour le post-traitement pour un gain d'exactitude moyen de 7,6% ce qui est relativement coûteux d'un point de vue exactitude/rendement. Cependant, sur les 26,77 s secondes utilisés au post-traitement, 24,20 s sont requises uniquement pour la case du prénom. Ce phénomène s'explique principalement par la méthode de post-traitement choisie

se basant sur la comparaison avec un dictionnaire. Comme il existe un nombre important de prénoms, que chacun des prénoms à plusieurs orthographes possibles, qu'il existe aussi plusieurs combinaisons de prénoms composés différents et ce dans deux langues différentes (anglais et français), le nombre de possibilités augmente rapidement. Seulement dans le cadre de cette expérience, le dictionnaire utilisé contenait une liste de 422 958 prénoms québécois. Ainsi, afin de déterminer la correction à suggérer, il est nécessaire de comparer et calculer la distance de Levenshtein pour chacune de ces combinaisons jusqu'à trouver celle qui permet le minimum de modifications. De plus, comme l'architecture présentée dans le chapitre 2 ne distingue pas la lettre « I » de la lettre « L », il est requis de réaliser cette opération en considérant une lettre « I » et puis avec la lettre « L » et de conserver le choix avec la distance minimale ce qui multiplie le nombre de combinaisons possibles. Ainsi, la décision prise initialement pour augmenter le taux d'exactitude s'avère plus coûteuse sur le temps de calculs comme il est nécessaire de vérifier l'ensemble des combinaisons.

À titre comparatif, le dictionnaire utilisé pour les noms de famille contenait uniquement 1004 noms ce qui justifie un traitement beaucoup plus rapide (0,32 s). On remarque ainsi une faiblesse dans l'utilisation du dictionnaire au niveau du temps de traitement lorsque le nombre de cas possibles n'est pas suffisamment réduit au préalable. Cependant, le temps de traitement obtenu semble suffisamment proche dans ce scénario pour considérer que l'objectif est atteint et que de simples optimisations dans la programmation permettront de s'assurer de respecter le seuil. Différentes pistes de solutions afin d'améliorer l'exactitude et le temps de traitement seront abordées à la sous-section suivante.

Tableau 11 : Performances sur quatre formulaires structurés avec caractères typographiques disponible à l'annexe III. La première colonne représente le numéro de case dans le formulaire pour un ordre de haut en bas et de gauche à droite. Le contenu attendu dans la case est décrit dans la seconde colonne. Le temps de reconnaissance requis (incluant le prétraitement) et le temps de post-traitement sont ensuite présentés aux colonnes trois et quatre sont additionnés à la colonne cinq. Les colonnes six et sept permettent de comparer le taux d'exactitude obtenu avant et après post-traitement. Les lignes avec un taux d'exactitude inférieur à l'objectif de 85% sont mises en évidence en jaune.

Case ID	Contenu	Temps requis reconnaissance (s)	Temps moyen requis correction (s)	Temps total (s)	Exactitude moyenne sans correction (%)	Exactitude moyenne avec correction (%)
1	family_name	7,14	0,32	7,46	90,83	100,00
2	given_name	5,35	24,20	29,54	96,43	100,00
3	middle_name	0,00	0,00	0,00	N/A	N/A
4	year_of_birth	3,00	0,00	3,00	68,75	93,75
5	month_of_birth	1,50	0,00	1,50	62,50	100,00
6	day_of_birth	1,50	0,00	1,50	87,50	100,00
7	gender_male	0,00	0,00	0,00	100,00	100,00
8	gender_female	0,00	0,00	0,00	100,00	100,00
9	gender_X	0,00	0,00	0,00	100,00	100,00
10	service_number	6,55	0,00	6,55	91,67	91,67
11	rank	1,21	0,00	1,21	37,50	37,50
12	language_EN	0,00	0,00	0,00	100,00	100,00
13	language_FR	0,00	0,00	0,00	100,00	100,00
14	number_and_street	15,65	0,23	15,88	91,96	96,29
15	apt./unit	1,36	0,00	1,36	27,78	27,78
16	city	6,88	0,35	7,23	86,46	100,00
17	province	8,06	0,17	8,23	80,83	100,00
18	postal_code	4,98	0,00	4,98	91,67	91,67
19	lot_and_concession_number	0,00	0,00	0,00	N/A	N/A
20	section	0,00	0,00	0,00	N/A	N/A
21	same_ordinary_address	0,00	0,00	0,00	100,00	100,00
22	mailing_address	0,00	0,00	0,00	N/A	N/A
23	mailing_country	0,00	0,00	0,00	N/A	N/A
24	mailing_province	0,00	0,00	0,00	N/A	N/A
25	mailing_city	0,00	0,00	0,00	N/A	N/A
26	mailing_postal_code	0,00	0,00	0,00	N/A	N/A
27	mailing_address_unit	15,22	0,72	15,94	72,98	77,84
28	mailing_country_unit	5,79	0,15	5,94	91,67	100,00
29	mailing_province_unit	8,24	0,20	8,44	82,02	100,00
30	mailing_city_unit	8,44	0,43	8,87	76,39	100,00
31	mailing_postal_code_unit	4,93	0,00	4,93	83,33	83,33
32	personal_information	0,00	0,00	0,00	100,00	100,00
33	signature	0,00	0,00	0,00	100,00	100,00
Total		105,78	26,77	132,55	84,81	91,99

3.2.2 Pistes d'améliorations

Comme mentionné précédemment, la principale faiblesse du système sur des formulaires contenant des caractères typographiques se trouve principalement entre la confusion entre certaines combinaisons de caractères. Ce phénomène peut facilement s'expliquer par le fait que les réseaux de neurones utilisés n'ont vu qu'une seule image de chacun de caractères typographiques lors de l'entraînement. En effet, comme l'apprentissage était réalisé avec une rétropropagation à chaque échantillon de 32 images de manière aléatoire, les exemplaires uniques se trouvent forcément noyés dans le bruit des images des caractères manuscrits ce qui ne permet pas au réseau d'apprendre les caractères typographiques et de bien généraliser son apprentissage. Les caractères manuscrits étant aussi différents des caractères typographiques, l'apprentissage sur des caractères manuscrits ne permet pas nécessairement une généralisation adéquate pour les caractères typographiques. Afin de résoudre ce problème, l'une des approches qui pourrait être considérée serait de générer une base de données de caractères typographiques uniquement et de réaliser une augmentation des données en bruitant les images de ces caractères. Ainsi, de la même manière qu'il a été présenté lors du chapitre 2, de nouveaux réseaux de neurones pourront apprendre de cette nouvelle base de données sans présence du bruit d'images de caractères manuscrits.

Cependant, cette nouvelle proposition entraîne la problématique que deux architectures de réseaux de neurones, l'une pour les caractères manuscrits et l'autre pour les caractères typographiques doivent être utilisés. Dans le scénario où l'on ne connait pas le type d'écriture au préalable du formulaire, la méthode la plus directe serait de réaliser la reconnaissance avec les deux architectures de manière séparée, puis de conserver l'option contenant le moins d'erreurs ce qui nécessiterait en théorie approximativement le double du temps total mesuré au tableau 11, ce qui n'est que peu intéressant au niveau du rendement exactitude/temps de calcul pour n'augmenter l'exactitude que de quelques pourcents. Toutefois, cette idée pourrait être appliquée en se limitant à un nombre de cases réduit. Par exemple, la case du nom de famille et celle de l'année de naissance étant généralement rapides à traiter pourrait

servir afin de réaliser un essai après le prétraitement. Cet essai pourrait permettre d'extraire les deux cases mentionnées afin de comparer les résultats des deux architectures de réseaux de neurones. Dans le cas de figure où l'architecture de réseaux de neurones typographiques obtiendrait des résultats incohérents ou peu probables, on pourrait ainsi se douter que l'écriture dans le formulaire est manuscrite puisque cette architecture n'ayant jamais vu de caractères manuscrits aura nécessairement une mauvaise généralisation sur les caractères manuscrits beaucoup plus complexes. De cette manière, il est possible d'utiliser l'architecture manuscrite sur l'ensemble du formulaire directement. À l'inverse, dans le cas de figure où les deux ont des résultats comparables ou que l'architecture typographique offre un résultat supérieur, cette architecture pourra être utilisée sur le reste du formulaire. Cette idée serait à approfondir et valider lors d'essais futurs, mais devrait permettre d'améliorer l'exactitude sur les caractères typographiques sans négliger de manière trop importante le temps de traitement. Dans le même ordre d'idées, si une architecture spécialisée à ce type d'écriture permet d'améliorer suffisamment le taux d'exactitude pour atteindre un taux près de 100%, certaines étapes du post-traitement pourraient être ignorées afin d'accélérer le traitement tel que le dictionnaire sur les prénoms.

3.3 PERFORMANCES DU SYSTEME SUR FORMULAIRE AVEC DES CARACTERES MANUSCRITS

3.3.1 Analyse des résultats

À partir des différents formulaires obtenus et leurs résultats associés se trouvant à l'annexe IV, un tableau cumulatif contenant la moyenne des résultats pour chacune des cases est présenté au tableau 12 pour des formulaires avec des caractères manuscrits. En jaune sont mis en évidences les cases avec une exactitude après post-traitement inférieure au minimum de 85% fixé comme objectif lors de l'introduction générale. De ce tableau, les six (6) derniers formulaires de l'annexe ont été exclus (pages 144 à 155) dû à trois (3) problèmes qui réduisent de manière conséquente l'exactitude et le temps de traitement et qui rendent

difficiles toutes comparaisons avec les résultats précédents vu l'impact non-négligeable de ces problèmes sur les performances globales.

Parmi les formulaires rejetés, trois (3) d'entre-eux (pages 146 à 151) l'ont été par l'utilisation de l'écriture cursive partiellement ou totalement dont l'étude a été repoussée à un projet de recherche futur comme détaillé lors du chapitre 4. Toutefois, comme des pistes de solutions sont tout de même présentées dans ce dernier chapitre, le système d'OCR actuel a tout de même été utilisé afin d'aider à identifier les problèmes et ainsi orienter la recherche. Ainsi, le huitième formulaire, le neuvième formulaire et le dixième formulaire de l'annexe ont été exclus des résultats cumulatifs au tableau 12. Le taux d'exactitude moyen avec correction de chacun de ces formulaires est dans l'ordre, 69,42%, 48,87% et 53,10% avec des temps de traitement totaux de 247,61 s, 882,85 s et 612,53 s. On remarque ainsi qu'autant en termes de taux d'exactitude qu'en temps de traitement, ces trois formulaires sont nettement en dessous des objectifs visés ce qui est attendu puisque l'une des principales hypothèses était que chacun des caractères pouvait être traité séparément ce qui n'est pas valable pour de l'écriture cursive. Ainsi, dans le cas de l'écriture cursive, la segmentation réalisée dans le formulaire repose en grande partie sur le système de fenêtrage présenté à la sous-section 2.2.2.1 comme cas particulier. Cependant, cette méthode n'ayant pas été développée plus en détails, celle-ci ne permet pas de reconnaître adéquatement des cas plus complexes et encore moins dans des délais courts dû au nombre important de calculs et d'images à générer et classifier. En somme, ces résultats demeurent tout de même intéressants puisqu'ils illustrent que la méthode peut tout de même réussir à identifier partiellement des caractères, mais que celle-ci est limitée dans son fonctionnement actuel.

Parmi les six (6) rejets, on retrouve aussi les formulaires rédigés au crayon plomb (pages 150 à 155) dont un qui était déjà rejeté dû à l'écriture cursive. Il est possible de voir que dans ces trois cas, la numérisation a été moins bien réussie, et ce, en essayant l'ensemble des paramètres disponibles sur le numériseur. En pratique, une numérisation de cette qualité serait probablement rejetée par défaut puisque l'information est partiellement illisible pour le client et ne répond pas aux normes de qualités. Cependant, comme pour le problème

précédent, ceux-ci ont tout de même été traités par le système afin d'identifier des points à améliorer. Ce problème de numérisation a pour conséquence d'entraîner de grandes variances dans l'intensité du trait pouvant aller de traits pâles à très foncés. Cependant, comme mentionné lors du chapitre 2, la méthode Otsu utilisée pour le seuillage est beaucoup plus efficace lorsque le trait est uniforme et que le contraste est élevé comme l'histogramme est plus facile à diviser adéquatement. Ainsi, le seuillage inadapté sur les formulaires au crayon plomb a tendance à sur-segmenter les caractères en plusieurs morceaux ce qui amènent plusieurs faux caractères introduits dans la reconnaissance. Une partie du problème est corrigée par la méthode de fusion des cases présentée dans les cas particuliers pour la segmentation à la sous-section 2.2.2.1, plusieurs cas persistent et ajoutent ainsi plusieurs sources d'erreurs difficiles à corriger. C'est pourquoi ces trois formulaires ont un taux d'exactitude moyen après correction dans l'ordre de 53,10%, 76,15% et 65,49% respectivement, soit beaucoup plus faible que le minimum fixé. Ceux-ci ont été traités en 612,53 s, 140,25 s et 329,97 s ce qui est plus élevé que la moyenne des formulaires.

Le dernier rejet concerne le septième formulaire écrit au stylo vert (pages 144 et 145) et le problème se produit au niveau de la segmentation. En effet, lors de la conception du système d'OCR, une hypothèse générale a été considérée que le texte se trouvait entièrement dans la case, ou du moins en proportion suffisante pour ne pas influencer la reconnaissance. Cependant, dans le cas du formulaire écrit au stylo vert, plusieurs caractères dépassent des cases partiellement ou totalement ce qui entraîne plusieurs erreurs de détection. Principalement, des caractères tronqués, en plus d'être beaucoup plus difficile à reconnaître, peuvent parfois créer plusieurs contours différents entraînant selon l'endroit de la troncature de la sur-segmentation. Dans le cas de ce formulaire, un taux d'exactitude moyen avec correction de 66,58% a été atteint en 483,00 s.

Tableau 12 : Performances sur six formulaires structurés avec caractères manuscrits disponible à l'annexe IV. La première colonne représente le numéro de case dans le formulaire pour un ordre de haut en bas et de gauche à droite. Le contenu attendu dans la case est décrit dans la seconde colonne. Le temps de reconnaissance requis (incluant le prétraitement), le temps de passé à glisser des fenêtres pour segmenter les caractères et le temps de post-traitement sont ensuite présentés aux colonnes trois, quatre et cinq sont additionnés à la colonne six. Les colonnes sept et huit permettent de comparer le taux d'exactitude obtenu avant et après post-traitement. Les lignes avec un taux d'exactitude inférieur à l'objectif de 85% sont mises en évidence en jaune.

Case ID	Contenu	Temps requis reconnaissance (s)	Temps moyen fenêtrage (s)	Temps moyen requis correction (s)	Temps total (s)	Exactitude moyenne sans correction (%)	Exactitude moyenne avec correction (%)
1	family_name	9,71	2,77	0,29	9,99	67,64	91,67
2	given_name	7,84	2,13	27,00	34,84	81,35	88,89
3	middle_name	1,56	1,41	1,54	3,10	N/A	N/A
4	year_of_birth	2,74	0,00	0,00	2,74	95,83	100,00
5	month_of_birth	1,43	0,00	0,00	1,43	100,00	100,00
6	day_of_birth	1,40	0,00	0,00	1,40	100,00	100,00
7	gender_male	0,00	0,00	0,00	0,00	100,00	100,00
8	gender_female	0,00	0,00	0,00	0,00	100,00	100,00
9	gender_X	0,00	0,00	0,00	0,00	100,00	100,00
10	service_number	11,88	6,17	0,00	11,88	88,89	88,89
11	rank	1,04	0,00	0,00	1,04	93,33	93,33
12	language_EN	0,00	0,00	0,00	0,00	100,00	100,00
13	language_FR	0,00	0,00	0,00	0,00	100,00	100,00
14	number_and_street	19,69	4,90	0,78	20,47	75,49	81,18
15	apt,/unit	0,50	0,00	0,00	0,50	100,00	100,00
16	city	12,68	5,30	0,34	13,03	80,98	100,00
17	province	14,44	8,76	0,14	14,58	76,98	100,00
18	postal_code	4,64	0,00	0,00	4,64	75,00	75,00
19	lot_and_concession_number	0,73	0,00	0,00	0,73	100,00	100,00
20	section	1,50	0,00	0,00	1,50	60,00	60,00
21	same_ordinary_address	0,00	0,00	0,00	0,00	100,00	100,00
22	mailing_address	0,00	0,00	0,00	0,00	N/A	N/A
23	mailing_country	0,00	0,00	0,00	0,00	N/A	N/A
24	mailing_province	0,00	0,00	0,00	0,00	N/A	N/A
25	mailing_city	0,00	0,00	0,00	0,00	N/A	N/A
26	mailing_postal_code	0,00	0,00	0,00	0,00	N/A	N/A
27	mailing_address_unit	10,46	3,38	0,79	11,24	82,53	88,45
28	mailing_country_unit	8,02	5,23	0,07	8,09	83,33	100,00
29	mailing_province_unit	11,16	5,95	0,06	11,22	69,05	100,00
30	mailing_city_unit	3,95	0,00	0,14	4,09	83,33	100,00
31	mailing_postal_code_unit	2,68	0,00	0,00	2,68	77,78	77,78
32	personal_information	0,00	0,00	0,00	0,00	100,00	100,00
33	signature	0,00	0,00	0,00	0,00	100,00	100,00
Total		128,03	46,00	31,14	159,18	88,57	94,27

Si l'on se concentre sur les six premiers formulaires de l'annexe IV (pages 132 à 143) qui ont été conservés et compilés au tableau 12, il est possible d'observer un taux d'exactitude moyen avec correction de 94,27%. On remarque ici qu'encore une fois, l'objectif de 85% fixé dans l'introduction générale a été atteint. De plus, le taux d'exactitude moyen sans correction est supérieur à celui présenté pour de l'écriture typographique au tableau 11. Ce résultat est encore une fois cohérent puisque les réseaux de neurones ont presqu'exclusivement été entraînés sur de l'écriture manuscrite, amenant donc de meilleurs résultats sur ces cas. Toutefois, quatre (4) cases obtiennent un taux d'exactitude moyen inférieur au seuil fixé, soit l'adresse, le code postal (x2) et la section.

Au niveau de l'adresse, l'un des principaux problèmes identifiés sur cette case est encore une fois lié à la grande flexibilité de cette case. En effet, comme celle-ci peut contenir une combinaison inconnue de lettres et de chiffres, cela favorise l'apparition de certaines confusions lors de la classification entres des chiffres et des caractères qui se ressemblent tel un « 1 » et un « l ». De plus, cette case ne contient actuellement qu'un post-traitement léger dans lequel on tente de séparer le numéro du reste de l'adresse ce qui ne permet pas de corriger des erreurs dans le nom de la rue ou de la route par exemple. De plus, les caractères spéciaux (« @ », « - », « ' », etc.) n'ont pas été ajoutés dans l'architecture des réseaux neuronaux dû à l'absence d'un nombre suffisant d'exemples pour permettre un entraînement, même si ceux-ci peuvent apparaître sur certains formulaires. Par conséquent, ces caractères sont systématiquement mal classifiés. Le trait d'union, l'apostrophe et l'arobase notamment sont des caractères spéciaux que l'on retrouve plus fréquemment dans les formulaires observés c'est pourquoi ceux-ci pourraient être intégrés à une future base d'apprentissage.

Concernant les codes postaux, le principal problème provient du fait que la case est relativement petite pour le nombre de caractères à y entrer. Cela a pour effet que certains caractères sont généralement moins bien tracés et beaucoup plus proches les uns des autres jusqu'à se toucher. Certains caractères vont même déborder de la case et entraîner ainsi des troncatures sur ces caractères. Par conséquent, on retrouve des problèmes de sous-segmentations, de sur-segmentation ainsi que des erreurs de classification.

Pour la section, le problème est relativement similaire à celui de l'adresse. En effet, la case pouvant contenir des lettres ainsi que des chiffres, cela favorise l'apparition de confusions qui sont éliminées dans des cas plus restreints.

Au niveau du temps total, le système prend en moyenne 159,18 s pour 27 cases (environ 5,90 s/case) ce qui est plus lent que le résultat observé pour les formulaires typographiques au tableau 11. Cependant, ce phénomène s'explique simplement par le fait que, pour les formulaires typographiques, les caractères sont bien découpés ce qui facilite grandement le traitement. Dans le cas de formulaires manuscrits, certains caractères peuvent se toucher ce qui requiert une segmentation plus complexe avec la méthode de fenêtrage ce qui demande nécessairement plus de temps que si cette étape n'était pas nécessaire.

3.3.2 Pistes d'améliorations

Au niveau des formulaires rejetés, on remarque principalement des faiblesses au niveau de la segmentation pour l'écriture cursive, le seuillage pour les formulaires écrits au plomb ainsi que la segmentation pour les caractères débordant des cases. Chacune de ces faiblesses représente ainsi une bonne piste d'amélioration pour améliorer le système.

Au niveau du seuillage, même si la méthode Otsu offre de bons résultats dans la majorité des cas, une autre méthode pourrait être envisagée. En effet, la méthode Otsu est beaucoup moins performante lorsque l'intensité du trait est variable puisqu'il est plus difficile de sélectionner un seuil à partir de l'histogramme de l'image. Ce problème a amené à utiliser d'abord un seuil adaptatif local sur la case afin de vérifier si la case est vide, mais aussi à sur-segmenter lorsque le trait du crayon n'est pas uniforme. Ainsi, il pourrait être envisagé d'approfondir avec d'autres méthodes proposées par (Cheriet, Kharma, Liu, & Suen, 2007; Gupta, Jacobson, & Garcia, 2007; Jyotsna, Chauhan, Sharma, & Doegar, 2016).

Au niveau de la segmentation des caractères débordant des cases, l'une des principales solutions au problème serait de chercher la position du texte à partir de l'image du formulaire

auquel le modèle a été soustrait. De cette manière, au lieu de rechercher spécifiquement le contenu se limitant à une case, il serait possible de rechercher des groupes de caractères par dilation sur l'ensemble de la page. Toutefois, cette méthode est sensible aux erreurs d'alignement et d'inclinaison entre le modèle et le formulaire puisqu'une erreur d'un ou deux pixels génère du bruit lors de la soustraction qui amènerait de faux positifs. Ainsi, il pourrait être pertinent pour investiguer des alternatives à la méthode *ORB*. Finalement, le fenêtrage demeure toujours une faiblesse du présent système dû à la difficulté de bien segmenter des caractères qui se touchent dans un délai raisonnable. Cette méthode serait donc aussi à investiguer plus en profondeur pour améliorer le système d'OCR. En attendant ces améliorations, le système demeure parfaitement fonctionnel et l'utilisation d'un chien de garde limitant le temps de traitement d'un formulaire peut être utilisé pour rejeter les formulaires problématiques comme ceux-ci prennent généralement beaucoup plus de temps et ainsi laisser un opérateur s'occuper des cas complexes.

3.4 CONCLUSION

Pour conclure, le présent chapitre a permis d'évaluer le taux d'exactitude ainsi que le temps de traitement du système d'OCR à l'aide d'un formulaire structuré réel, et ce, pour une écriture typographique et une écriture manuscrite. Dans le cas de l'écriture typographique, le taux d'exactitude moyen après post-traitement est de 91,99% et les 25 cases ont nécessité en moyenne un temps de traitement total de 132,55 s (4,02 s/case). L'objectif principal fixé dans le cadre de l'introduction générale est ainsi validé dans le cadre de formulaires structurés avec une écriture typographique. Les erreurs de détection présentes après post-traitement pour la plupart des formulaires sont le résultat de confusions entre des caractères avec une forte ressemblance comme le chiffre « 1 » et la lettre « l ». Afin de résoudre ce problème, l'une des approches qui pourrait être considérée serait de se générer une base de données de caractères typographiques uniquement et de réaliser une augmentation des données en bruitant les images de ces caractères. Concernant le temps de traitement, celui-ci pourra

principalement être amélioré en optimisant la programmation comme le temps de traitement mesuré se trouve à la limite de l'objectif fixé.

Dans le cas de l'écriture manuscrite, le taux d'exactitude moyen après post-traitement est de 94,27% et les 27 cases ont nécessité en moyenne un temps de traitement total de 159,18 s (5,90 s/case). L'objectif principal fixé dans le cadre de l'introduction générale est ainsi validé dans le cadre de formulaires structurés avec une écriture manuscrite au niveau de l'exactitude, mais le système s'avère plus lent que l'objectif initialement fixé (4 s/case) principalement à cause de l'ajout de la méthode de fenêtrage qui est coûteuse en temps de calculs. Plusieurs formulaires ont aussi été rejetés de l'échantillon, on remarque principalement des faiblesses au niveau de la segmentation pour l'écriture cursive, le seuillage pour les formulaires écrits au plomb ainsi que la segmentation pour les caractères débordant des cases. Chacune de ces faiblesses représente ainsi une bonne piste d'amélioration pour améliorer le système. Cependant, le système peut tout de même être utilisé pour traiter les cas les plus simples à l'aide d'un chien de garde limitant le temps de traitement et rejetant les formulaires plus complexes en attendant les améliorations.

CHAPITRE 4
SYSTÈME DE RECONNAISSANCE SUR DOCUMENTS NON-STRUCTURÉS

4.1 INTRODUCTION

Ce chapitre présente une piste de solutions pour le fonctionnement du système de reconnaissance de caractères réalisé pour des formulaires non-structurés tels que celui présenté à l'annexe II. Ainsi, pour chacun des sous-systèmes identifiés et des connaissances acquises lors des chapitres deux et trois ainsi que les schématisations réalisées aux figures 6 et 7, une approche est présentée pour traiter des documents non-structurés. Cependant, cette approche n'a pas été ni traitée ni codée ni expérimentée, ces différentes étapes pourront être réalisées dans des travaux futurs.

Contrairement à ce qui a été établi dans le cas de formulaires structurés, les différentes étapes de prétraitement et de segmentation ne peuvent reposer sur la structure du document ou sur un modèle ce qui complexifie de manière importante le problème.

Par exemple, la méthode *ORB* était utilisée lors du chapitre deux afin de calculer la matrice homographique et ainsi corriger l'alignement et l'inclinaison du texte. Cependant, n'ayant pas de modèle de référence dans le cas de formulaires non-structurés comme la nature des documents varie, cet algorithme n'est ici plus applicable pour corriger l'alignement. De plus, comme le texte ne se trouve pas nécessairement dans des cases définies, celui-ci peut se trouver à n'importe quel endroit dans l'image et avoir une inclinaison variable dans une même phrase ou un même mot comme illustré à la figure 40.

Figure 40 : Exemple d'écriture à inclinaison variable

Dans le même ordre d'idée, il n'existe pas de type de filtrage ou de seuillage qui est performant pour tous types de documents. Certains articles comme *OCR binarization and image pre-processing for searching historical documents* (Gupta, Jacobson, & Garcia, 2007) et *Binarization Techniques for Degraded Document Images – A Review* (Jyotsna, Chauhan, Sharma, & Doegar, 2016) montrent que la méthode Otsu performe plutôt bien pour des documents historiques ce qui est aussi cohérent avec les résultats obtenus pour les formulaires structurés. Cependant, comme mentionné dans le chapitre deux, la méthode Otsu peut surestimer ou sous-estimer grandement le seuillage. Au niveau de la segmentation, un problème supplémentaire présent dans ce type de document est l'écriture cursive. Comme vu précédemment, la détection de contours tient pour acquis que les caractères sont tous bien distincts. En pratique, même dans des formulaires structurés, si certains caractères se touchaient, ceux-ci étaient difficiles à segmenter de manière rapide et efficace. Ainsi, la méthode utilisée précédemment pour détecter les contours ne semble pas être la plus adaptée pour segmenter des mots en écriture cursive tel que présenté à la figure 40.

On remarque ainsi plusieurs problèmes dans le programme développé précédemment lorsque l'on souhaite appliquer celui-ci pour des documents non-structurés. Plusieurs des méthodes retenues profitaient de caractéristiques propres aux formulaires qui permettaient de simplifier l'approche et réduire la variabilité des paramètres à ajuster. Cependant, dans le cas de formulaires non-structurés, ces approximations et ces hypothèses générales ne sont plus valables, ce qui rend difficile d'adapter ces méthodes et leurs paramètres sans perdre en capacité de généralisation.

4.2 DETECTION DU TEXTE, CORRECTION DE L'ALIGNEMENT ET DE L'INCLINAISON ET SEGMENTATION

Dans la revue de littérature, certaines méthodes proposaient soit d'utiliser une série de transformations morphologiques et la transformée de Hough ou des méthodes plus avancées à base de réseaux de neurones afin de détecter le texte et corriger l'alignement (Baviskar, Ahirrao, Potdar, & Kotecha, 2021; Cheriet, Kharma, Liu, & Suen, 2007). Toutefois, lors des essais sur des formulaires structurés, il a été possible de constater qu'il est difficile de déterminer les transformations morphologiques adéquates pour tous types de documents. En effet, comme la position et la taille de l'écriture ainsi que les espacements varient, un nombre d'opérations différentes sont nécessaires pour obtenir des performances optimales sur chacun des formulaires ce qui ne permet pas d'avoir une solution simple d'utilisation et viable à long terme pour la *DIORG*. Concernant les approches reposant sur les réseaux de neurones, les articles *Start, Follow, Read : End-to-End Full-Page Handwriting Recognition* (Wigington, et al., 2018) et *Building an efficient OCR system for historical documents with little training data* (Martínek, Lenc, & Král, 2020) permettent de montrer des résultats satisfaisants sans information préalable sur la structure du document.

La méthode proposée dans le cadre de formulaires structurés s'inspire ainsi directement des deux articles mentionnés ci-dessus ainsi que des observations réalisées dans le cadre du système sur des formulaires structurés. La principale méthode qui serait à investiguer dans une série d'expérimentation serait de réaliser la détection du texte et la correction de l'alignement et de l'inclinaison avec un *Region Proposal Network* (RPN). Comme mentionné auparavant, lorsque l'on parle de formulaire non-structuré, la position du texte est inconnue au préalable. Ainsi, un RPN permettrait de balayer le document afin de déterminer des régions d'intérêts pouvant contenir du texte. À la figure 41 est illustré un exemple de détection d'un paragraphe comme région d'intérêt. Cette méthode de balayage pouvant être coûteuse en temps de calcul, l'utilisation d'une architecture de type CNN serait à étudier puisque cette architecture permet de réduire le nombre d'opérations en comparaison à un

réseau à connectivité totale et ainsi obtenir une estimation rapide de la position du texte dans l'image.

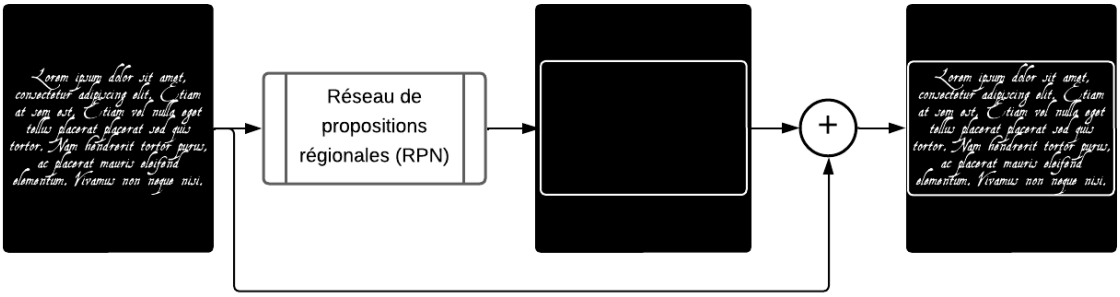

Figure 41 : Exemple d'utilisation d'un RPN afin de déterminer les différentes régions d'intérêts pouvant contenir de l'écriture dans un document non-structuré. À partir de la région obtenue, un réseau de neurones pour détecter le début de la ligne et un réseau de neurones pour suivre la ligne peuvent être utilisés afin de corriger l'alignement et l'inclinaison d'un texte manuscrit.

Une fois la position approximative détectée, il est alors possible de tenter de détecter l'alignement et l'inclinaison des différentes zones de texte avant l'extraction des caractéristiques. Pour ce faire, pour chacune des lignes de zones de textes, une méthode similaire à l'article *Start, Follow, Read : End-to-End Full-Page Handwriting Recognition* (Wigington, et al., 2018) pourrait être envisagée. Cette méthode présentée dans l'article consiste à tenter de détecter les coordonnées de début de ligne à l'aide d'un réseau de neurones pour ensuite suivre la ligne avec un autre réseau de neurones permettant ainsi de déterminer les coordonnées, l'échelle et la rotation nécessaire pour chacune des régions détectées. Cette même approche pourrait être utilisée sur les régions proposées par le RPN et ainsi déterminer l'ensemble des paramètres pour corriger l'alignement et l'inclinaison. Dans l'article, l'auteur arrive, sur certains exemples, à segmenter des lignes de textes pouvant être courbées ou avec des inclinaisons variables comme à la figure 40 et en faire la correction au niveau de la ligne, ce qui semble intéressant pour traiter une plus grande variété de documents. De plus, une fois la ligne de texte segmentée, il est alors possible de séparer les

mots à l'aide des espacements en se basant sur une détection de contours comme pour les formulaires structurés. Toutefois, la méthode proposée par l'article demande un temps de calcul qui sera typiquement supérieur dû à la nécessité d'utiliser deux réseaux de neurones pour détecter le début des lignes et suivre le texte. En combinaison avec le réseau RPN, cela pourrait avoir un effet non-négligeable sur la performance rendant la solution non viable pour une numérisation à plus grande échelle. Il serait ainsi nécessaire de déterminer le temps de calcul associé pour certains documents pour ce type de solution et chercher à réduire le nombre d'opérations nécessaires le cas échéant en plus d'en valider le taux d'exactitude avant d'implanter une telle solution.

Cette approche combinée avec l'utilisation d'un RPN pourrait cependant permettre de détecter le texte, segmenter les lignes, corriger l'alignement et l'inclinaison puis segmenter de nouveau les lignes en mots avec aucune information préalable sur le document. De plus, l'utilisation de réseaux de neurones dans le processus permet de réduire significativement le nombre d'étapes de prétraitement nécessaires (filtrage et seuillage) comme il est possible de réaliser un apprentissage sur les données brutes réduisant encore une fois le nombre de paramètres à ajuster manuellement. Or, cette approche requiert un nombre important de données étiquetées si l'on souhaite utiliser directement les données brutes puisque l'on ne réduit pas la dimension de l'espace (fléau de la dimension) ce qui avait pu être significativement réduit dans le cas de formulaires structurés à l'aide du prétraitement et des hypothèses (Bishop, 2006). Ce besoin en grand volume de données ainsi que l'échéancier fixé pour la réalisation du projet sont d'ailleurs les principales causes qui ont amené le projet à se limiter à présenter une piste de solution pour la résolution des documents non-structurés. Ce problème pourrait devenir moins significatif avec le temps puisque la *DIORG* réalise actuellement déjà un étiquetage sur un grand volume de données manuellement chaque année dans leurs tâches quotidiennes. Ainsi, en conservant ces données accessibles pour un entraînement, il devrait être possible d'envisager certaines de ces nouvelles pour les documents non-structurés.

4.3 EXTRACTION DES CARACTERISTIQUES ET CLASSIFICATION D'IMAGES

Au niveau de la classification des images, deux possibilités seraient à investiguer dans le cas de formulaires non-structurés :

- La première possibilité consiste à vérifier si les mêmes réseaux de neurones entraînés pour des formulaires structurés permettent d'obtenir des résultats satisfaisants à l'aide de la méthode discuté à la sous-section précédente et en glissant une fenêtre sur les mots segmentés (de la même manière que pour des lettres collées). Avec un nombre suffisant de données ajoutées à la base de données pour des formulaires non-structurés, il devrait techniquement être possible d'apprendre aux réseaux à traiter les formulaires structurés et non-structurés et ainsi éviter d'avoir à réaliser une nouvelle architecture pour les documents non-structurés réduisant le temps de développement et facilitant l'entretien du système. Cependant, cette possibilité possède plusieurs faiblesses difficiles à quantifier sans vérifier le tout en conditions réelles. Dans un premier temps, l'ajout de nouvelles données de formulaires non-structurés pourrait potentiellement nuire aux apprentissages sur des formulaires structurés et réduire les performances dans ces cas. En effet, comme les caractères typographiques et cursifs ont des formes différentes, il est possible que l'ajout de nouvelles données provenant d'écriture cursive ne fasse que confondre les différents réseaux ayant pour conséquence de réduire ainsi la performance pour les formulaires structurés. De plus, l'architecture réalisée s'avère assez lourde en termes de calculs vu la grande quantité de réseaux pour les lettres et pour les chiffres. Ce problème pouvait être minimisé en connaissant le contenu attendu d'une case pour des formulaires structurés, mais dans le cas de formulaire non-structurés, considérer 184 réseaux en parallèle avec un RPN, un réseau pour détecter les lignes et un réseau pour suivre les lignes ne permettrait pas de respecter les délais d'une minute par document pour des documents contenant plusieurs phrases. Il faudrait donc déterminer la validité de l'approche et

analyser les performances d'un tel système lorsqu'un nombre suffisant de données auront été accumulées pour permettre un apprentissage de cette architecture.

- La seconde possibilité serait de tester différentes architectures de réseaux afin de réaliser l'extraction des caractéristiques et la reconnaissance des caractères. Plusieurs pistes de solutions sont ici possibles. Comme mentionné dans l'article *Handwritten Optical Character Recognition (OCR) : A Comprehensive Systematic Literature Review (SLR)* (Memon, Sami, Khan, & Uddin, 2020), les architectures que l'on retrouve le plus souvent dans la littérature et qui obtiennent les meilleures performances sont actuellement les CNN, les réseaux de neurones récurrents (RNN) et les réseaux récurrents à mémoire court et long terme (LSTM). Comme on ne traitait que des caractères individuels, les réseaux de type CNN n'avaient pas été considérés puisqu'un réseau à connectivité total permet théoriquement de ne rejeter aucune solution et les images étaient suffisamment petites pour réduire l'intérêt principal des CNN qui permettent de réduire le temps de calcul sur des images en favorisant des groupes de pixels locaux. Les réseaux de type RNN et LSTM avaient aussi moins d'intérêt dans l'hypothèse où l'on classifie les images une par une puisque l'on ne profite pas de la capacité de mémoire de ces réseaux à leur plein potentiel n'ayant pas nécessairement de continuité entre les caractères individuels. Toutefois, si l'on considère que pour traiter les caractères cursifs, il est possible de faire glisser une fenêtre sur le mot entier comme proposé plus haut, chacune des images présentées aux réseaux ont un chevauchement entre elles, et donc une continuité qui pourrait être exploitée par un réseau de type RNN ou LSTM. Par conséquent, il pourrait être intéressant de tenter d'utiliser l'un de ces deux types de réseaux pour réaliser la classification. Il pourrait aussi être intéressant de vérifier si les caractéristiques extraites pour les formulaires structurés demeurent pertinentes avec une nouvelle architecture. Finalement, certaines architectures de type « hybride » combinent des réseaux de type RNN ou LSTM

avec un réseau de type CNN afin de simplifier les traitements et accélérer le temps de calcul. Il s'agit ainsi d'une option qui pourrait s'avérer intéressante dans une éventuelle continuité du projet afin d'évaluer les performances de chacune de ces options.

4.4 CONCLUSION

Pour conclure, le présent chapitre, en complément au chapitre 1, a permis de présenter une piste de solutions se basant sur une combinaison d'un RPN, un LSTM et un CNN pour réaliser la reconnaissance optique de caractères manuscrits sur des documents non-structurés. Comme il n'est généralement pas possible de poser des hypothèses générales préalables vu la variété de la nature des documents, plusieurs décisions prises dans le cadre des formulaires structurés n'étaient ici plus applicables ou moins adaptées. La nouvelle solution proposée repose sur la littérature et consiste en un RPN pour détecter le texte, un réseau pour détecter le début des lignes avec un réseau permettant de suivre les lignes ainsi qu'une extraction des caractéristiques et la classification de fenêtres extraites à l'aide d'un LSTM combiné avec un CNN. Toutefois, le manque de données présentement disponible au sein de la *DIORG* pour réaliser un entraînement ne permet pas de mettre à l'essai cette option à court terme pour en vérifier la validité et comparer les performances. Ainsi, cette proposition agit à titre de première piste de solutions pour une future recherche et devra être analysée de nouveau selon les nouveaux avancements théoriques lors d'un nouveau projet.

CONCLUSION GÉNÉRALE

L'objectif de ce projet de recherche est de concevoir un système de reconnaissance de caractères manuscrits sur des formulaires fédéraux et tester ses performances dans une mise en situation réelle pour une implémentation dans les processus de la *DIORG* afin de résoudre les problématiques de préservation, de retranscription, d'accessibilité et de sécurité que rencontre le gouvernement du Canada dans sa gestion documentaire. Le taux d'exactitude visé dans le cas de formulaires est de 85% et un formulaire d'une quinzaine de cases doit être traité en une minute ou moins. À partir des résultats obtenus, le second objectif est de proposer des pistes pour un développement sur des documents historiques qui permettrait un taux d'exactitude de l'ordre de 75% avec un temps de calcul comparable à celui du premier objectif. Ces deux principaux objectifs ont aussi permis d'identifier une série d'objectifs secondaires :

1. Avoir un système complet incluant une interface pour l'entrée, le traitement et la sortie des données ;
2. Avoir un système facile d'opération pour une personne non spécialisée ;
3. Avoir un système donnant le niveau de confiance de la validité des résultats ;
4. Avoir un système avec une interface permettant de revérifier facilement les résultats avec le niveau de confiance le plus faible ;
5. Avoir un système qui s'adapte facilement à de nouveaux formulaires ou documents ;
6. Avoir un système évolutif permettant un autoapprentissage des formes d'écriture.

Afin d'atteindre ces objectifs, les différentes étapes composant un système de reconnaissance optique des caractères ont été étudiées en consultant la littérature ainsi que la documentation de logiciels commerciaux. De cette étude, il a été possible d'établir qu'un tel système était constitué des étapes suivantes : prétraitement, segmentation, extraction des caractéristiques, classification des images et post-traitement. Quatre hypothèses générales ont principalement été posées lors de la réalisation du système :

1. Le modèle du formulaire est disponible ;
2. Il n'y a pas d'autres déformations ou distorsions optiques/géométriques dans l'image du formulaire outre l'alignement et l'inclinaison :
3. Le texte dans un formulaire se trouve à l'intérieur des espaces désignés ;
4. Les caractères peuvent être considérés comme des contours distincts.

À partir de ces différentes informations, il a été possible de développer un système de reconnaissance optique des caractères sur des formulaires structurés. Au niveau du prétraitement, la méthode *ORB* a été utilisée afin d'aligner le formulaire numérisé avec le modèle permettant ainsi de corriger les problèmes d'alignement et d'inclinaison. Au niveau du seuillage, un seuil adaptatif local est utilisé afin d'estimer si du texte est présent dans la case et la méthode d'Otsu est appliquée sur les cases non vides.

Au niveau de la segmentation, une première détection de contours est utilisée après une série de dilation afin d'estimer la position du texte dans la case. Par la suite, une seconde détection de contours est réalisée sur la région identifiée afin de segmenter les différentes lettres. Lors de certains cas particuliers, une fusion de contours ainsi qu'une méthode de fenêtrage peuvent être employées afin d'améliorer la segmentation.

L'une des principales contributions de ce projet de recherche se trouve au niveau de l'extraction des caractéristiques et à la classification des images. La transformée en cosinus discrète (DCT) est employée pour une caractérisation fréquentielle, la transformée de Hough pour une caractérisation spatiale et la transformée en ondelettes discrète (DWT) pour une

caractérisation fréquentielle et spatiale de l'image en plus de l'image brute. Pour classifier l'ensemble de ces données, un total de 190 réseaux de neurones ont été entraînés dont 188 selon une approche « Un contre tous ». De ce nombre total, 40 réseaux de neurones (4 par classes pour chacune des caractérisations) sont utilisés pour reconnaître les 10 chiffres (0 à 9), alors que 144 sont utilisés pour reconnaitre les lettres (4 par classes où certaines lettres ont deux représentations selon la case). Quatre réseaux de neurones sont aussi utilisés afin de distinguer les caractères entre les chiffres et les lettres pour des cases contenant de l'information mixte. Finalement, les deux derniers réseaux de neurones permettent d'utiliser l'ensemble des sorties pour les 40 réseaux de neurones pour les chiffres ou les 144 réseaux de neurones pour les lettres afin de considérer l'ensemble des poids et des dynamiques apprises pour établir une prédiction finale.

Concernant le post-traitement, une combinaison de dictionnaires, de restrictions selon les cases et de contexte est utilisée pour corriger les erreurs de détection. De plus, une architecture de système avec une phase d'apprentissage répétée sur de nouvelles données est suggérée pour permettre au système d'améliorer ses prédictions et profiter du grand volume de données que traite la *DIORG*.

Le système de reconnaissance conçu a, par la suite, été mis à l'essai sur 16 exemplaires fictifs d'un formulaire structuré réel actuellement traité à la *DIORG*. Sur ces 16 formulaires, quatre (4) ont été remplis à l'ordinateur avec une écriture typographique, alors que 12 ont été remplis par trois (3) employés de manière manuscrite puis numérisés. Sur des formulaires structurés avec une écriture typographique, le taux d'exactitude moyen après post-traitement est de 91,99%, il faut en moyenne 4,02 s pour traiter une case et la principale erreur est la confusion entre des caractères similaires tels que le chiffre 1 avec le caractère « l ». Pour une écriture manuscrite, le taux d'exactitude après post-traitement est de 94,27% et il faut en moyenne 5,90 s pour traiter une case atteignant ainsi les objectifs fixés en termes de taux d'exactitude. Les principales faiblesses se trouvent au niveau de la segmentation de l'écriture cursive, du seuillage lorsqu'il y a de grandes variations d'intensité dans le trait et sur la segmentation lorsque des caractères sortent des cases.

Au niveau des documents historiques non structurés, l'ensemble des connaissances acquises lors de la conception du système a permis d'établir que les principaux défis se trouvent au niveau de la détection du texte, la correction de l'alignement et de l'inclinaison ainsi que dans la segmentation de l'écriture cursive. Une solution à base de réseau de propositions régionales (RPN), de réseau de neurones convolutifs (CNN) et de réseaux de neurones récurrents (RNN) est suggérée afin de pallier certaines faiblesses observées.

Pour conclure, ce travail de recherche a permis d'atteindre les objectifs. Cependant, le système développé est présentement limité à des formulaires de types structurés contenant une écriture manuscrite ou typographique constituée des caractères découpés et formée d'un trait uniforme et continu. Les caractères reconnus se limitent présentement aux lettres (A à Z) sans diacritiques ainsi qu'aux chiffres. De plus, plusieurs améliorations demeurent à apporter. Au niveau du système actuel, le développement d'une architecture séparée pour les caractères typographiques, la méthode de seuillage pour des traits variant d'intensité et le raffinement de la méthode de segmentation et plus particulièrement du fenêtrage sont à envisager. Au niveau des documents historiques non structurés, le problème demeure quant à lui ouvert principalement au niveau de la détection du texte, de la correction de l'alignement et de l'inclinaison et de la segmentation de l'écriture cursive. De manière plus exhaustive, les travaux futurs qui pourraient être envisagés à la suite de cette recherche sont les suivants :

- Déterminer et implanter une méthode permettant d'améliorer le seuillage sur des caractères à intensité variable ;
- Déterminer une méthode pour détecter les caractères sortant des cases ;
- Étendre la plage de reconnaissance aux caractères avec diacritiques (« é », « è », « ê », etc.) ainsi que les caractères spéciaux (« @ », « - », « ‘ », etc.) ;
- Déterminer, implanter et quantifier des méthodes de détection et de correction d'erreurs additionnelles ;

- Quantifier l’effet sur les performances de l’usage du système à apprentissage répété proposé ;
- Concevoir une architecture de réseaux de neurones pour les caractères typographiques et déterminer une manière efficace pour détecter le type d’écriture (typographique ou manuscrit) d’un formulaire ;
- Concevoir et quantifier les performances d’un système permettant de traiter des formulaires non-structurés et l’écriture cursive.

ANNEXES

ANNEXE I : FORMULAIRE STRUCTURÉ

Travaux publics et Services gouvernementaux Canada | Public Works and Government Services Canada

439-F

Protégé « B » lorsque rempli

FORMULAIRE DU RÉGIME DE SERVICES DENTAIRES POUR LES PENSIONNÉS (RSDP) - 1er avril 2006

Le RSDP est offert à la plupart des pensionnés de la fonction publique par le gouvernement du Canada. En remplissant et en signant le formulaire, vous acceptez l'offre de participation au RSDP. Une fois le formulaire rempli, veuillez l'envoyer à votre conseiller en rénumération ou au bureau de pension.

La communication des renseignements demandés dans ce document est facultative. Ces renseignements sont recueillis aux fins d'appliquer le Règlement du RSDP et sont essentiels pour assurer la protection demandée. Un refus de remplir ce formulaire peut vous exposer à une application rejetée ou retardée. Ces renseignements seront versés au fichier de renseignements personnels TPSGC PCE 702. Ils sont protégés contre toute divulgation à des personnes ou à des organismes non autorisés, conformément aux dispositions de la *Loi sur la protection des renseignements personnels*. Aux termes de ladite loi, vous avez le droit de vous faire communiquer les renseignements personnels vous concernant et conservés par une institution du gouvernement fédéral et de demander des corrections si, selon vous, ils sont erronés ou incomplets. Les renseignements personnels que vous fournissez au sujet d'une autre personne peuvent être communiqués à celle-ci en vertu de la *Loi sur la protection des renseignements personnels*.

PARTIE A - À ÊTRE REMPLIE PAR LE CONSEILLER EN RÉMUNÉRATION OU PAR LE BUREAU DU RÉGIME DE PENSION

☐ Demande initiale ☐ Modification ☑ Annulation — **5 - Annulation volontaire (permanente)**

John	Coltrane		
350 ch. du Jazz		N° d'app. 2	N° de téléphone - Domicile (418 524 1350
Ville/Village BluesTown	Province/État QC		Français
Pays Canada	Code postal/Code de zone G0J 1S0		

Régime de pension : LPRGRC — Loi sur la pension de retraite de la Gendarmerie royale du Canada (LPRGRC)

Pensionné à titre de : Étudiant - S1

PARTIE B - À ÊTRE REMPLIE PAR LE PENSIONNÉ

Acceptation de l'offre de participation au régime - Je certifie que les membres de ma famille énumés ci-dessous, ainsi que moi-même, répondent aux conditions d'admissibilité du RSDP et je choisi une des catégories de protection indiquées ci-dessous.

☐ **Catégorie I** Pensionné seulement

☑ **Catégorie II** Pensionné et un membre admissible de sa famille

☐ **Catégorie III** Pensionné et plus d'un membre admissible de sa famille

Membres de ma famille à être protégés :

Prénoms du (de la) conjoint(e)/ conjoint(e) de fait	Nom de famille du (de la) conjoint(e)/conjoint(e) de fait	Date de naissance A M J
Paul	McCartney	2001 05 25

☒ Conjoint(e) — Date du mariage: A M J 2019 01 04

☐ Conjoint(e) de fait — Date de début/fin de cohabitation: A M J

Prénoms du membre de la famille	Nom du membre de la famille	Date de naissance A M J

☐ Enfant admissible moins de 21 ans ☐ Étudiant admissible entre 21 et 25 ans ☐ Enfant admissible souffrant de déficience ☐ Enfant admissible adopté de fait

Prénoms du membre de la famille	Nom du membre de la famille	Date de naissance A M J

☐ Enfant admissible moins de 21 ans ☐ Étudiant admissible entre 21 et 25 ans ☐ Enfant admissible souffrant de déficience ☐ Enfant admissible adopté de fait

☑ J'ai attaché une feuille séparée avec les noms, les dates de naissance et le lien de parenté des membres de famille additionnels.

En signant ce formulaire, je suis conscient et j'ai lu et comprends les raisons pour la collection de l'information personnelle et l'engagement sur la page 2. Je suis d'accord avec les conditions associées au RSDP. J'autorise toute institution gouvernementale ou agence à donner à l'administrateur du RSDP et à Travaux publics et Services gouvernementaux Canada l'information requise pour vérifier les renseignements fournis sur ce formulaire, pour compléter mon inscription au RSDP, et pour administrer le RSDP.

2021-10-30 — Date

[signature] — Signature

Travaux publics et Services gouvernementaux Canada | Public Works and Government Services Canada

439-F

Protégé « B » lorsque rempli

N° de pension 000925316

FORMULAIRE DU RÉGIME DE SERVICES DENTAIRES POUR LES PENSIONNÉS (RSDP) - 1er avril 2006

PARTIE C - À ÊTRE REMPLIE PAR LE BUREAU DU RÉGIME DE PENSION

Date d'entrée en vigueur de la retenue (début-modification-arrêt) A M J	Nom de l'agent de pension (en lettres moulées S.V.P.)	Numéro de téléphone Code régional
20210307	Billy Jeans	514 2343796
Date d'entrée en vigueur de la protection (début-modification-arrêt) A M J	**Signature**	**Date A M J**
20210307		2021 10 31

ENGAGEMENT

Ce formulaire d'adhésion inclut les dispositions du RSDP, y compris toutes les modalités et conditions, comme si elles avaient été imprimées sur ce formulaire. En signant ce formulaire et en le renvoyant à mon conseiller en rémunération ou à mon administrateur de régime de pension, ce formulaire constitue un accord entre le gouvernement du Canada et moi-même au sujet de mon adhésion au RSDP et son application à mon égard. Je conviens que les dispositions du RSDP et de l'accord peuvent être modifiés par le gouvernement du Canada. Le RSDP et l'accord modifiés s'appliqueront alors comme s' ils avaient été imprimés sur ce formulaire. Je comprends que les taux de cotisation peuvent changer, tel que déterminé par le président du Conseil du Trésor du Canada.

Je ne pourrai mettre fin à mon adhésion ou celle des membres éligibles de ma famille dans le RSDP qu'après une période de trois années civiles révolues. J'autorise les retenues mensuelles requises sur ma pension autant pour les fins du présent régime que pour les taxes applicables.

NOTES EXPLICATIVES

1. Le livret d'information sur l'adhésion présente un résumé du RSDP. Ce livret qui s'intitule Régime de services dentaires pour les pensionnés - Renseignements sur l'adhésion et sommaire du régime (pour les pensionnés s'inscrivant le ou après le 1er avril 2006), ne renferme cependant pas toutes les dispositions du régime.

2. Le site Web du Secrétariat du Conseil du Trésor du Canada situé à l'adresse suivante : http://www.tbs-sct.gc.ca renferme le texte complet du RSDP et du livret d'information. Vous pouvez également obtenir un exemplaire de ces documents en communiquant avec le Centre de distribution du Secrétariat du Conseil du Trésor du Canada par téléphone au (613) 995-2855 ou par courriel à Services-Distribution@tbs-sct.gc.ca et en donnant les numéros de document TBS 006779 pour le Règlement du RSDP et TBS 006796 pour le livret d'information.

3. Si vous n'avez pas suffisament d'espace pour identifier tout les membres de la famille que vous voulez protéger, veuillez indiquer leur nom, leur lien de parenté avec vous, et leur date de naissance sur une feuille de papier distincte et joindre celle-ci au présent formulaire.

4. Pour les résidants de l'Ontario et du Québec, la taxe de vente provinciale s'ajoute au taux de cotisation. De plus, les résidants du Québec devront peut-être acquitter l'impôt sur le revenu du Québec sur l'avantage imposable (voir le livret d'information sur l'adhésion).

5. En règle générale, la protection en vertu du RSDP entre en vigueur le premier jour du deuxième mois suivant le mois au cours duquel le bureau chargé des pensions pertinent reçoit le formulaire du RSDP dûment rempli. Toutefois, les nouveaux pensionnés qui soumettent un formulaire rempli dans les 60 jours suivant la date d'entrée en vigueur de leur droit à pension seront normalement protégés en vertu du RSDP à compter de la date d'entrée en vigueur de leur droit à pension.

6. Si le membre admissible de votre famille est un enfant admissible souffrant une déficience ou un enfant admissible adopté de fait, les pieces justificatives doivent être fournies.

Date de réception au bureau des pensions

ANNEXE II : FORMULAIRE NON-STRUCTURÉ

DEPARTMENT OF MARINE, CANADA

Form 37A — Meteorological Service of Canada

HEAD OFFICE TORONTO, ONT.

Station Meadow Lake Prov. Sask. Month Oct. 1934 Observer C. Morton

Western Observers give Lot ..... Section 34 Tp. 19 Range 21 W. of 2nd Mer. Eastern give Lot ..... Con. ..... Tp. ..... Co. .....

DAY OF MONTH	TEMPERATURE Max.	Min.	Min. on grass	PRECIPITATION RAIN p.m.	RAIN next a.m.	SNOW p.m.	SNOW next a.m.	Total 24 hours	CLOUDINESS Amt. of Cloud 0-10 a.m.	p.m.	Mean
	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
1	61	33	28								
2	62	24	38								
3	62	42	20		.10			.10			
4	59	32	27	.12	.10			.22			
5	55	44	11								
6	70	27	43								
7	72	30	42								
8	70	39	31								
9	78	38	40								
10	83	38	45								
11 ✓	88	35	53 ✓								
12 ✓	88	43	45								
13	78	35	43								
14	55	32	23								
15	61	29	32								
16	55	35	20								
17	46	15	31								
18	52	30	22								
19	58	33	25								
20	67	24	43								
21	64	40	24								
22	62	43	19								
23	56	23	33								
24	48	23	25		.04			.04			
25	46	24	22								
26	44	26	18								
27 ✓	45	07	38								
28	43	27	16								
29	37	25	12								
30	36	25	11								
31	30	10	20								
Sums	1831	931	900					.36			
Means	59.1	30.0	29.1								

OBSERVER—Note Total Depth of Snow on Ground or Soil Moisture each Monday morning, and last day of month.

Date Mon.	Mon.	Mon.	Mon.	Mon.	End of Month
Depth or Moisture					

THIS SPACE FOR OFFICE USE ONLY.

Week Ending	Mean t.	Diff.	Precip.	Diff.	Cloud	Diff.	Grass
1							
2							
3							
4							
5							

WEEKLY TOTALS AND MEANS

Every Monday morning after obs. take totals of last seven entries in the columns indicated. To get weekly mean divide totals by 7.

First week ending Mon. the 1st

FROM COL.	TOTAL	MEAN	
(2)	357	(a) 51	Weekly Mean Max.
(3)	203	(b) 29	Weekly Mean Min.
(4)		(c)	Mean Grass Min.
(9)	(d)	X	Weekly Precip.
(12)		(e)	Weekly Cloudiness

Second week ending Mon. the 8th

(2)	479	(a) 67 1/7	Weekly Mean Max.
(3)	238	(b) 34	Weekly Mean Min.
(4)		(c)	Mean Grass Min.
(9)	(d)	X	Weekly Precip.
(12)		(e)	Weekly Cloudiness.

Third week ending Mon. the 15th

(2)	531	(a) 75 5/7	Weekly Mean Max.
(3)	253	(b) 35 4/7	Weekly Mean Min.
(4)		(c)	Mean Grass Min.
(9)	(d)	X	Weekly Precip.
(12)		(e)	Weekly Cloudiness

Fourth week ending Mon. the 22nd

(2)	404	(a) 57 5/7	Weekly Mean Max.
(3)	~~220~~	(b) 31 3/7	Weekly Mean Min.
(4)		(c)	Mean Grass Min.
(9)	(d)	X	Weekly Precip.
(12)		(e)	Weekly Cloudiness.

Fifth week ending Mon. the 29th

(2)	319	(a) 45 4/7	Weekly Mean Max.
(3)	153	(b) 22 1/7	Weekly Mean Min.
(4)		(c)	Mean Grass Min.
(9)	(d)	X	Weekly Precip.
(12)		(e)	Weekly Cloudiness

MONTHLY TOTALS—Means, Diff., Extremes.

Mean Max.	Diff.	Mean Min.	Diff.	Grass	Diff.	Precip.	Diff.
59.1		30.0				.36	

Mean Temp.	Diff.	Max.	Date	Min.	Date
44.6		88	11+12	7	27th

Grass Min.	Date	Mean Range	Extreme Range
		29.1	53

No. of Days' Rain	Snow	Total	0-.09"	.10-.19"	.20-.29"	.30-.39"	.40-.49"
3	0	3	1	1	1		

ANNEXE III : FORMULAIRES STRUCTURÉS AVEC CARACTÈRES TYPOGRAPHIQUES POUR ÉVALUATION DES PERFORMANCES ET RÉSULTATS

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Garand	Antoine	

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1919 / 10 / 20	Male ☐ Female ☐ Gender X ☑	844 512 784	2	English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street
40359 Tantalus Way

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
	Squamish	British Columbia	V0N 1T0

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☑

Mailing address

City, town, village or municipality	Province/territory/state	Country	Postal Code

Unit mailing address

Current mailing address
2301 Rue de l'Église

City, town, village or municipality	Province/territory/state	Country	Postal code
Val-David	Quebec	Canada	J0T 2N0

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☑

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X _____ _____
Signature of Canadian Forces elector *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code _____

ED name _____

Unit election officer signature _____ Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	GARAND	6,21	1	('GARAND', 0)	0,29	0,00	Garand	100,00	100,00	6,50
2	given_name	1	ANLOLNE	6,40	1	('ANTOINE', 1)	39,11	0,00	Antoine	85,71	100,00	45,51
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	7979	3,08	2	1979	0,00	0,00	1919	50,00	75,00	3,08
5	month_of_birth	0	70	1,46	2	IMPOSSIBLE VALUE	0,00	0,00	10	50,00	N/A	1,46
6	day_of_birth	0	20	1,55	2	20	0,00	0,00	20	100,00	100,00	1,55
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
10	service_number	0	844512784	6,74	0	844512784	0,00	0,00	844512784	100,00	100,00	6,74
11	rank	2	2	0,71	0	2	0,00	0,00	2	100,00	100,00	0,71
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	4035GTANT2LUSWAY	17,86	3	40359 TANTALUSWAY	0,92	0,00	40359 Tantalus Way	87,50	100,00	18,78
15	apt./unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	SQUAMLSN	7,33	1	('SQUAMISH', 1)	0,39	0,00	Squamish	87,50	100,00	7,72
17	province	1	BRLTLSNCULUMBLA	13,73	1	('BRITISH COLUMBIA', 4)	0,18	0,00	British Columbia	73,33	100,00	13,91
18	postal_code	3	V0N7T0	4,99	0	V0N7T0	0,00	0,00	V0N 1T0	83,33	83,33	4,99
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	2307RUNDBLLGLLSE	16,69	3	2307 RUNDBLLGLLSE	0,00	0,00	2301 rue de l'Église	64,71	64,71	16,69
28	mailing_country_unit	1	CANADA	5,67	1	('CANADA', 0)	0,14	0,00	Canada	100,00	100,00	5,81
29	mailing_province_unit	1	AUEBEC	5,67	3	('QUEBEC', 1)	0,18	0,00	Québec	83,33	100,00	5,85
30	mailing_city_unit	1	VALDAVLD	7,33	3	('VAL-DAVID', 2)	0,40	0,00	Val-David	77,78	100,00	7,73
31	mailing_postal_code_unit	3	J0T2N0	4,95	3	J0T2N0	0,00	0,00	J0T 2N0	100,00	100,00	4,95
32	personal_information	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
33	signature	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
Total				110,37			41,61	0,00		89,30	96,65	151,98

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Desjardins	Joseph	

Date of birth (yyyy-mm-dd)			Gender	Service number (mandatory)	Rank	Language
1982	12	12	Male ☑ Female ☐ Gender X ☐	378 651 268	11	English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street
170 Metcalfe St

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
102	Ottawa	Ontario	K2P 1P3

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☑

Mailing address

City, town, village or municipality	Province/territory/state	Country	Postal Code

Unit mailing address

Current mailing address
300 Wallinger Ave

City, town, village or municipality	Province/territory/state	Country	Postal code
Kimberley	British Columbia	Canada	V1A 1Z4

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☑

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X _____ _____
Signature of Canadian Forces elector *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code _____

ED name _____

Unit election officer signature _____ Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	DEGARDLNS	8,97	1	('DESJARDINS', 2)	0,37	0,00	Desjardins	80,00	100,00	9,34
2	given_name	1	JOSEPH	5,49	1	('JOSEPH', 0)	15,87	0,00	Joseph	100,00	100,00	21,36
3	middle_name	1		0,01	1	NONE	0	0,00	NONE	N/A	N/A	0,01
4	year_of_birth	0	7982	2,98	2	1982	0	0,00	1982	75,00	100,00	2,98
5	month_of_birth	0	72	1,4	2	IMPOSSIBLE VALUE	0	0,00	12	50,00	N/A	1,40
6	day_of_birth	0	72	1,45	2	IMPOSSIBLE VALUE	0	0,00	12	50,00	N/A	1,45
7	gender_male	4	[True]	0	0	[True]	0	0,00	True	100,00	100,00	0,00
8	gender_female	4	[False]	0	0	[False]	0	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0	0	[False]	0	0,00	False	100,00	100,00	0,00
10	service_number	0	378657268	6,49	0	378657268	0	0,00	378651268	88,89	88,89	6,49
11	rank	2	T7	1,6	0	T7	0	0,00	11	0,00	0,00	1,60
12	language_EN	4	[True]	0	0	[True]	0	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0	0	[False]	0	0,00	False	100,00	100,00	0,00
14	number_and_street	2	170METCALFEST	12,33	3	7 70METCALFEST	0	0,00	170 Metcalfe St	100,00	92,31	12,33
15	apt./unit	2	707	2,19	3	707	0	0,00	102	16,67	16,67	2,19
16	city	1	OTTBWA	5,48	1	('OTTAWA', 1)	0,35	0,00	Ottawa	83,33	100,00	5,83
17	province	1	ONTARLO	6,48	1	('ONTARIO', 0)	0,18	0,00	Ontario	100,00	100,00	6,66
18	postal_code	3	K2P1P3	5	0	K2P1P3	0	0,00	K2P 1P3	100,00	100,00	5,00
19	ot_and_concession_numbe	2		0	0	NONE	0	0,00	NONE	N/A	N/A	0,00
20	section	2		0	3	NONE	0	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0	0	[True]	0	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0	3	NONE	0	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0	1	NONE	0	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0	3	NONE	0	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0	3	NONE	0	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0	3	NONE	0	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	300WA111NGERAVE	14,62	3	300 WA111NGERAVE	0	0,00	300 Wallinger Ave	80,00	80,00	14,62
28	mailing_country_unit	1	CANBDB	5,69	1	('CANADA', 2)	0,16	0,00	Canada	66,67	100,00	5,85
29	mailing_province_unit	1	BRLLLSNCOLUMBLA	14,25	3	('BRITISH COLUMBIA', 4)	0,18	0,00	British Columbia	73,33	100,00	14,43
30	mailing_city_unit	1	KLMBERLEY	8,08	3	('KIMBERLEY', 1)	0,43	0,00	Kimberley	88,89	100,00	8,51
31	mailing_postal_code_unit	3	V7A7Z4	4,89	3	V7A7Z4	0	0,00	V1A 1Z4	66,67	66,67	4,89
32	personal_information	4	[True]	0	0	[True]	0	0,00	True	100,00	100,00	0,00
33	signature	4	[False]	0	0	[False]	0	0,00	False	100,00	100,00	0,00
Total				107,40			17,54	0,00		80,78	88,89	124,94

Elections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
Dumont	Lukas	

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
1948 / 05 / 18	Masculin ☐ Féminin ☐ Genre X ☐	789 452 156	4	Français ☑ Anglais ☐

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue
8815 Avenue du Parc

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
402	Montréal	Québec	H2N 1Y7

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : ☑

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale
6788 Thorold Stone Rd

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal
Niagara Falls	Ontario	Canada	L2J 1B4

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : ☐

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

X _____ _____
Signature de l'électeur des Forces canadiennes *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription _____

Nom de la circinscription _____

Signature du fonctionnaire électoral de l'unité _____ Date: A A A A M M J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	DUMONL	6,07	1	('DUMONT', 1)	0,32	0,00	Dumont	83,33	100,00	6,39
2	given_name	1	LUKAS	4,85	1	('LUKAS', 0)	16,80	0,00	Lukas	100,00	100,00	21,65
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	7948	3,00	2	1948	0,00	0,00	1948	75,00	100,00	3,00
5	month_of_birth	0	5	1,65	2	5	0,00	0,00	5	100,00	100,00	1,65
6	day_of_birth	0	18	1,45	2	18	0,00	0,00	18	100,00	100,00	1,45
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	789452156	6,55	0	789452156	0,00	0,00	789452156	100,00	100,00	6,55
11	rank	2	A	0,94	0	A	0,00	0,00	4	0,00	0,00	0,94
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	BB15AVENUEDUPARC	17,40	3	8815 AVENUEDUPARC	0,00	0,00	8815 Avenue du Parc	87,50	100,00	17,40
15	apt./unit	2	D02	2,33	3	D02	0,00	0,00	402	66,67	66,67	2,33
16	city	1	MONLRCAL	7,42	1	('MONTREAL', 2)	0,43	0,00	Montréal	75,00	100,00	7,85
17	province	1	AUEMC	4,65	1	('QUEBEC', 3)	0,14	0,00	Québec	50,00	100,00	4,79
18	postal_code	3	H2N7Y7	4,99	0	H2N7Y7	0,00	0,00	H2N 1Y7	83,33	83,33	4,99
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	6788TH0R0LDSTONERD	18,39	3	6788 THOROLDSTONERD	1,95	0,00	6788 Thorold Stone Rd	88,89	100,00	20,34
28	mailing_country_unit	1	CANADA	5,97	1	('CANADA', 0)	0,14	0,00	Canada	100,00	100,00	6,11
29	mailing_province_unit	1	ONLARLO	6,60	3	('ONTARIO', 1)	0,18	0,00	Ontario	85,71	100,00	6,78
30	mailing_city_unit	1	NLAGARAFALLS	10,97	3	('NIAGARA FALLS', 2)	0,51	0,00	Niagara Falls	83,33	100,00	11,48
31	mailing_postal_code_unit	3	L2J7B4	5,00	3	L2J7B4	0,00	0,00	L2J 1B4	83,33	83,33	5,00
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
Total				108,23			20,47	0,00		86,48	93,33	128,7

Elections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
Marceau	Simon	

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
1980 / 12 / 25	Masculin ☐ Féminin ☐ Genre X ☐	189 465 231	02	Français ☑ Anglais ☐

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue
1200 Waverley St

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
9	Winnipeg	Manitoba	R3T 0P4

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : ☑

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale
1866 Scugog St

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal
Port Perry	Ontario	Canada	L9L 1J3

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : ☐

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

X _____ _____
Signature de l'électeur des Forces canadiennes *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription _____

Nom de la circinscription _____

Signature du fonctionnaire électoral de l'unité _____ Date: A A A A M M J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	MARCEAU	7,30	1	('MARCEAU', 0)	0,31	0,00	Marceau	100,00	100,00	7,61
2	given_name	1	SIMON	4,64	1	('SIMON', 0)	25,01	0,00	Simon	100,00	100,00	29,65
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	7980	2,95	2	1980	0,00	0,00	1980	75,00	100,00	2,95
5	month_of_birth	0	72	1,48	2	IMPOSSIBLE VALUE	0,00	0,00	12	50,00	100,00	1,48
6	day_of_birth	0	25	1,54	2	25	0,00	0,00	25	100,00	100,00	1,54
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	789465237	6,41	0	789465237	0,00	0,00	189465231	77,78	77,78	6,41
11	rank	2	O2	1,59	0	O2	0,00	0,00	2	50,00	50,00	1,59
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	7200WAVERLEYST	15,01	3	7200 WAVERLEYST	0,00	0,00	1200 Waverley St	92,86	92,86	15,01
15	apt./unit	2	G	0,92	3	G	0,00	0,00	9	0,00	0,00	0,92
16	city	1	WLNNLPEG	7,29	1	('WINNIPEG', 0)	0,22	0,00	Winnipeg	100,00	100,00	7,51
17	province	1	MANLTOBA	7,39	1	('MANITOBA', 0)	0,18	0,00	Manitoba	100,00	100,00	7,57
18	postal_code	3	R3T0P4	4,92	0	R3T0P4	0,00	0,00	R3T 0P4	100,00	100,00	4,92
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	7B66SCUG0GSL	11,16	3	7 B66SCUGOGSL	0,94	0,00	1866 Scugog St	58,33	66,67	12,10
28	mailing_country_unit	1	CANADA	5,83	1	('CANADA', 0)	0,14	0,00	Canada	100,00	100,00	5,97
29	mailing_province_unit	1	ONLARLO	6,45	3	('ONTARIO', 1)	0,26	0,00	Ontario	85,71	100,00	6,71
30	mailing_city_unit	1	POAP?RRY	7,37	3	('PORT PERRY', 4)	0,39	0,00	Port Perry	55,56	100,00	7,76
31	mailing_postal_code_unit	3	L9L7J3	4,87	3	L9L7J3	0,00	0,00	L9L 1J3	83,33	83,33	4,87
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
Total				97,12			27,45	0,00		85,14	90,83	124,57

ANNEXE IV : FORMULAIRES STRUCTURÉS AVEC CARACTÈRES MANUSCRITS POUR ÉVALUATION DES PERFORMANCES ET RÉSULTATS

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Blier	Noëlla	ø

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1971 / 02 / 04	Male ☐ Female ☑ Gender X ☐	33 402 672	CR-03	English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street: 449 rue Blier

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
	Matane	Québec	G0J 3C2

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☑

Mailing address: 449 rue Blier

City, town, village or municipality	Province/territory/state	Country	Postal Code
Matane	Québec	Canada	G0J 3C2

Unit mailing address

Current mailing address: 449 rue Blier

City, town, village or municipality	Province/territory/state	Country	Postal code
Matane	Québec	Canada	G0J 3C2

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☐

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

x Novella Blier — *Signature of Canadian Forces elector*

2024-02-21 — *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code: ____

ED name: ____

Unit election officer signature: ____ Date: Y Y Y Y / M M / D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	BLLER	5,46	1	('BLIER', 1)	0,29	0,00	Blier	80,00	100,00	5,75
2	given_name	1	NODLLA	5,88	1	('NOELLA', 1)	32,46	0,00	Noella	83,33	100,00	38,34
3	middle_name	1	?	9,36	1	('A', 1)	9,22	8,45	NONE	N/A	N/A	18,58
4	year_of_birth	0	1971	2,68	2	1971	0,00	0,00	1971	100,00	100,00	2,68
5	month_of_birth	0	2	1,53	2	2	0,00	0,00	2	100,00	100,00	1,53
6	day_of_birth	0	4	1,42	2	4	0,00	0,00	4	100,00	100,00	1,42
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	33402672	5,61	0	33402672	0,00	0,00	33402672	100,00	100,00	5,61
11	rank	2	CRW03	4,04	0	CRW03	0,00	0,00	CR-03	80,00	80,00	4,04
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	449RUQBL1ER	10,36	3	44 9RUQBLLER	0,83	0,00	449 rue Blier	81,82	81,82	11,19
15	apt./unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	MATANE	5,49	1	('MATANE', 0)	0,24	0,00	Matane	100,00	100,00	5,73
17	province	1	QUQBEC	5,43	1	('QUEBEC', 1)	0,18	0,00	Québec	83,33	100,00	5,61
18	postal_code	3	G0J3C2	5,12	0	G0J3C2	0,00	0,00	G0J 3C2	100,00	100,00	5,12
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	449RUEBL1ER	10,45	3	44 9RUEBLLER	0,84	0,00	449 rue Blier	90,91	90,91	11,29
28	mailing_country_unit	1	LCANADA	6,42	1	('CANADA', 1)	0,18	0,00	Canada	83,33	100,00	6,60
29	mailing_province_unit	1	QUIBTC	5,65	3	('QUEBEC', 2)	0,17	0,00	Québec	66,67	100,00	5,82
30	mailing_city_unit	1	MATANE	5,52	3	('MATANE', 0)	0,24	0,00	Matane	100,00	100,00	5,76
31	mailing_postal_code_unit	3	G0I3C2	5,03	3	G0I3C2	0,00	0,00	G0J 3C2	83,33	83,33	5,03
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				95,45			44,65	8,45		93,03	97,34	140,1

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Aubin	Lise	

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1982 | 02 | 12	Male ☐ Female ☐ Gender X ☑	13924 7		English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street: 1821 avenue Crescent

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
33	ORANGE Country	Ontario	G1L 3F4

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☒

Mailing address

City, town, village or municipality	Province/territory/state	Country	Postal Code

Unit mailing address

Current mailing address

City, town, village or municipality	Province/territory/state	Country	Postal code

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☐

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X Lise Aubin — *Signature of Canadian Forces elector*

2024-02-21 — *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code: _____

ED name: _____

Unit election officer signature: _____ Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	AUBIK	4,76	1	('AUBIN', 1)	0,31	0,00	Aubin	80,00	100,00	5,07
2	given_name	1	LISE	3,51	1	('LISE', 0)	13,68	0,00	Lise	100,00	100,00	17,19
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	1982	2,86	2	1982	0,00	0,00	1982	100,00	100,00	2,86
5	month_of_birth	0	2	1,44	2	2	0,00	0,00	2	100,00	100,00	1,44
6	day_of_birth	0	12	1,58	2	12	0,00	0,00	12	100,00	100,00	1,58
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
10	service_number	0	139247	4,11	0	139247	0,00	0,00	139247	100,00	100,00	4,11
11	rank	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	L82LAVENUECRESCENT	16,40	3	1 82LAVENUECRESCENT	0,00	0,00	1821 avenue Crescent	88,89	94,44	16,40
15	apt./unit	2	33	1,41	3	33	0,00	0,00	33	100,00	100,00	1,41
16	city	1	ORKNGECBUNTRY	22,88	1	('ORANGE COUNTRY', 3)	0,56	11,88	Orange Country	76,92	100,00	23,44
17	province	1	ONTARIO	6,22	1	('ONTARIO', 0)	0,13	0,00	Ontario	100,00	100,00	6,35
18	postal_code	3	G1L3A	4,10	0	G1L3A	0,00	0,00	G1L 3F4	66,67	66,67	4,10
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2		0,01	3	NONE	0,00	0,00	NONE	N/A	N/A	0,01
28	mailing_country_unit	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
29	mailing_province_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
30	mailing_city_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
31	mailing_postal_code_unit	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				69,28			14,68	11,88		95,39	97,95	83,96

Elections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
COVINGTON	BRUCE	

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
2002 / 02 / 15	Masculin [X] Féminin [ ] Genre X [ ]	41 205 412		Français [ ] Anglais [X]

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue: 4001 RUE JOLIE-COEUR

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
2	GATINEAU	ONTARIO	G0Z 1Z4

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)
LOT 3	RANG 2

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : [X]

Adresse postale:

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale: 4001 RUE JOLIE-COEUR

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal
GATINEAU	ONTARIO	CANADA	G0Z 1Z4

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : [X]

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

X Bruce Covington — *Signature de l'électeur des Forces canadiennes*

2024-02-21 — *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription [ ]

Nom de la circinscription [ ]

Signature du fonctionnaire électoral de l'unité _____ Date: A A A A / M M / J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	POVLNGTOW	8,65	1	('COVINGTON', 2)	0,11	0,00	Covington	77,78	100,00	8,76
2	given_name	1	BRUCE	4,76	1	('BRUCE', 0)	4,01	0,00	Bruce	100,00	100,00	8,77
3	middle_name	1		0,01	1	NONE	0,00	0,00	NONE	N/A	N/A	0,01
4	year_of_birth	0	2002	3,11	2	2002	0,00	0,00	2002	100,00	100,00	3,11
5	month_of_birth	0	02	1,50	2	02	0,00	0,00	02	100,00	100,00	1,50
6	day_of_birth	0	15	1,49	2	15	0,00	0,00	15	100,00	100,00	1,49
7	gender_male	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	41205412	6,07	0	41205412	0,00	0,00	41205142	100,00	100,00	6,07
11	rank	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	400LRUEJ0LLEDC0EUR	17,50	3	4 00LRUEJOLLEDCOEUR	1,94	0,00	4001 rue Jolie-Cœur	72,22	83,33	19,44
15	apt,/unit	2	2	0,74	3	2	0,00	0,00	2	100,00	100,00	0,74
16	city	1	GATLNEAU	7,40	1	('GATINEAU', 0)	0,05	0,00	Gatineau	100,00	100,00	7,45
17	province	1	ONTARLO	6,44	1	('ONTARIO', 0)	0,01	0,00	Ontario	100,00	100,00	6,45
18	postal_code	3	G0Z1Z4	4,97	0	G0Z1Z4	0,00	0,00	G0Z 1Z4	100,00	100,00	4,97
19	lot_and_concession_number	2	LOT3	3,36	0	LOT3	0,00	0,00	Lot 3	100,00	100,00	3,36
20	section	2	RANGZ	4,75	3	RANGZ	0,00	0,00	Rang 2	80,00	80,00	4,75
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	4001RUEJ0LIEWC0EUR	16,28	3	4 001RUEJOLIEWCOEUR	2,07	0,00	4001 rue Jolie-Cœur	83,33	94,44	18,35
28	mailing_country_unit	1	CANADA	5,66	1	('CANADA', 0)	0,04	0,00	Canada	100,00	100,00	5,70
29	mailing_province_unit	1	OVTLAZIO	42,59	3	('ONTARIO', 3)	0,01	23,10	Ontario	57,14	100,00	42,60
30	mailing_city_unit	1	GATLNEAU	7,54	3	('GATINEAU', 0)	0,05	0,00	Gatineau	100,00	100,00	7,59
31	mailing_postal_code_unit	3	G0Z1Z4	5,42	3	G0Z1Z4	0,00	0,00	G0Z 1Z4	100,00	100,00	5,42
32	personal_information	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				148,24			8,29	23,10		95,02	98,38	156,53

Élections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
Bélanger	Darina	

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
1984 | 10 | 02	Masculin ☐ Féminin ☑ Genre X ☐	042 327 892		Français ☑ Anglais ☐

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue
24 route du village

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
	St-Louis	Québec	G9L 3R2

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : ☑

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : ☐

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

x Darina Bélanger — *Signature de l'électeur des Forces canadiennes*

2024-02-21 — *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription

Nom de la circinscription

Signature du fonctionnaire électoral de l'unité

Date: A A A A M M J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	TBLUSER	12,55	1	('TELLIER', 3)	0,33	6,51	Bélanger	25,00	50,00	12,88
2	given_name	1	DAPIQNA	6,32	1	('DAVINA', 2)	36,26	0,00	Darina	66,67	66,67	42,58
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	1984	2,70	2	1984	0,00	0,00	1984	100,00	100,00	2,70
5	month_of_birth	0	10	1,42	2	10	0,00	0,00	10	100,00	100,00	1,42
6	day_of_birth	0	2	1,35	2	2	0,00	0,00	2	100,00	100,00	1,35
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	041327892	6,23	0	041327892	0,00	0,00	042 327 892	88,89	88,89	6,23
11	rank	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	24R0IE?UV18LLYE	20,58	3	2 4RGIE?UV18LLYE	1,01	7,41	24 route du village	60,00	60,00	21,59
15	apt./unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	STWLOULE	7,33	1	('ST-LOUIS', 3)	0,40	0,00	St-Louis	62,50	100,00	7,73
17	province	1	QUEVBEC	6,41	1	('QUEBEC', 1)	0,18	0,00	Québec	83,33	100,00	6,59
18	postal_code	3	G4L3R2	4,56	0	G4L3R2	0,00	0,00	G4L 3R2	100,00	100,00	4,56
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
28	mailing_country_unit	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
29	mailing_province_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
30	mailing_city_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
31	mailing_postal_code_unit	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				69,45			38,18	13,92		88,13	92,53	107,63

Elections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
Bergeron	Céline	

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
2000 | 03 | 05	Masculin ☐ Féminin ☐ Genre X ☐	134 986 340	C	Français ☑ Anglais ☐

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue
1874 rue de l'avalanche

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
B	Warwick	Alberta	K1L 3V7

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)
	Rang 9

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : ☒

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : ☐

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

X Céline Bergeron — *Signature de l'électeur des Forces canadiennes*

2024-02-21 — *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription _____

Nom de la circinscription _____

Signature du fonctionnaire électoral de l'unité _____ Date: A A A A M M J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	BERAERON	7,77	1	('BERGERON', 1)	0,32	0,00	Bergeron	87,50	100,00	8,09
2	given_name	1	CEILIOHE	7,07	1	('CEILIDH', 2)	39,59	0,00	Céline	66,67	66,67	46,66
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	200	2,23	2	IMPOSSIBLE VALUE	0,00	0,00	2000	75,00	N/A	2,23
5	month_of_birth	0	3	1,36	2	3	0,00	0,00	3	100,00	100,00	1,36
6	day_of_birth	0	5	1,31	2	5	0,00	0,00	5	100,00	100,00	1,31
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	134286340	6,18	0	134286340	0,00	0,00	134 286 340	100,00	100,00	6,18
11	rank	2	C	0,85	0	C	0,00	0,00	C	100,00	100,00	0,85
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	L874RUELELLAVALARCLE	17,92	3	1 874RUELELLAVALARCLE	0,00	0,00	1874 rue de l'avalanche	75,00	80,00	17,92
15	apt./unit	2	B	0,87	3	B	0,00	0,00	B	100,00	100,00	0,87
16	city	1	?ARWICY	26,19	1	('WARWICK', 2)	0,41	19,94	Warwick	71,43	100,00	26,60
17	province	1	TL?RTN	17,88	1	('ALBERTA', 4)	0,18	12,82	Alberta	28,57	100,00	18,06
18	postal_code	3	K1L5V7	4,67	0	K1L5V7	0,00	0,00	K1L 3V7	83,33	83,33	4,67
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2	RQM9	3,41	3	RQM9	0,00	0,00	Rang 9	40,00	40,00	3,41
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
28	mailing_country_unit	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
29	mailing_province_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
30	mailing_city_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
31	mailing_postal_code_unit	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				97,71			40,50	32,76		87,02	93,50	138,21

Élections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
Martineau	Olivier	

Date de naissance (aaaa-mm-jj): 1986 | 12 | 11

Genre: Masculin [X] Féminin [ ] Genre X [ ]

Numéro matricule (obligatoire): 153 807 155

Grade: 03

Langue: Français [✓] Anglais [ ]

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue: 152 Route Centrale

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
	St-Ulric	Québec	G0J 3H0

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)
—	—

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : [✓]

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale: 303 rue Principale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal
Les Méchins	Québec	Canada	G0J 1T0

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : [ ]

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

X Olivier Martineau — *Signature de l'électeur des Forces canadiennes*

2024-02-21 — *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription: _____

Nom de la circinscription: _____

Signature du fonctionnaire électoral de l'unité: _____ Date: A A A A M M J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	?ARTLNDALL	19,05	1	('MARTINEAU', 4)	0,35	10,11	Martineau	55,56	100,00	19,40
2	given_name	1	CLIVIET	19,47	1	('OLIVIER', 2)	36,01	12,78	Olivier	71,43	100,00	55,48
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	1986	2,85	2	1986	0,00	0,00	1986	100,00	100,00	2,85
5	month_of_birth	0	12	1,31	2	12	0,00	0,00	12	100,00	100,00	1,31
6	day_of_birth	0	11	1,25	2	11	0,00	0,00	11	100,00	100,00	1,25
7	gender_male	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	153655715	43,06	0	153655715	0,00	37,01	153 807 155	44,44	44,44	43,06
11	rank	2	3	1,37	0	3	0,00	0,00	3	100,00	100,00	1,37
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	L52R0UTTLCENTRALE	35,37	3	1 52ROUTTLCENTRALE	0,92	22,01	152 route centrale	75,00	87,50	36,29
15	apt./unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	STWULRLC	6,81	1	('ST-ULRIC', 2)	0,39	0,00	St-Ulric	75,00	100,00	7,20
17	province	1	Q?BEC	44,23	1	('QUEBEC', 2)	0,17	39,72	Québec	66,67	100,00	44,40
18	postal_code	3	W3S4M	4,43	0	W3S4M	0,00	0,00	G0J 3H0	0,00	0,00	4,43
19	lot_and_concession_number	2	T	1,00	0	T	0,00	0,00	NONE	N/A	N/A	1,00
20	section	2	M	0,84	3	M	0,00	0,00	NONE	N/A	N/A	0,84
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	303R1IEPRLNC1QXXL?	35,99	3	3 03RLIEPRLNCLQXXL?	1,82	20,25	303 rue Principale	73,33	80,00	37,81
28	mailing_country_unit	1	?NADA	36,04	1	('CANADA', 2)	0,20	31,38	Canada	66,67	100,00	36,24
29	mailing_province_unit	1	QUETBEC	18,73	3	('QUEBEC', 1)	0,18	12,61	Québec	83,33	100,00	18,91
30	mailing_city_unit	1	LPSMEMCDHLNS	10,64	3	('LES MECHINS', 5)	0,52	0,00	Les méchins	50,00	100,00	11,16
31	mailing_postal_code_unit	3	C0O6T1M	5,63	3	C0O6T1M	0,00	0,00	G0J 1T0	50,00	50,00	5,63
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				288,07			40,56	185,87		79,64	90,08	328,63

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Paradis	Geneviève	

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1988 03 02	Male ☐ Female ☒ Gender X ☐	152 801 153	04	English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street: 854 rue Bailey

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
	Matane	Quebec	G4W 3T4

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)
—	—

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☑

Mailing address:

City, town, village or municipality	Province/territory/state	Country	Postal Code

Unit mailing address

Current mailing address: 4010 rue Belzile

City, town, village or municipality	Province/territory/state	Country	Postal code
Rimouski	Quebec	Canada	L6Z 3H0

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☐

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X Geneviève Paradis — *Signature of Canadian Forces elector*

2024-02-21 — *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code: ____

ED name: ____

Unit election officer signature: ____ Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	PARNDIXS	7,46	1	('PARADIS', 2)	0,32	0,00	Paradis	71,43	100,00	7,78
2	given_name	1	CWINQUBBJP	8,92	1	('CINDUJA', 5)	49,89	0,00	Geneviève	0,00	0,00	58,81
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	1988	2,80	2	1988	0,00	0,00	1988	100,00	100,00	2,80
5	month_of_birth	0	3	1,37	2	3	0,00	0,00	3	100,00	100,00	1,37
6	day_of_birth	0	2	1,40	2	2	0,00	0,00	2	100,00	100,00	1,40
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	172801133	18,25	0	172801133	0,00	11,94	152801153	77,78	77,78	18,25
11	rank	2	0U	1,64	0	0U	0,00	0,00	04	50,00	50,00	1,64
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	85URUPDBXJ?J	40,11	3	854 RUPDBXJ?J	0,00	28,46	854 rue Boulay	33,33	41,67	40,11
15	apt./unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	??INPT	48,04	1	('BRANT', 4)	0,42	0,00	Matane	0,00	0,00	48,46
17	province	1	?LDP?D	74,72	1	('NL', 5)	0,19	42,23	Québec	0,00	0,00	74,91
18	postal_code	3	R0Q1I1R7	6,25	0	R0Q1I1R7	0,00	69,46	G4W 3J4	0,00	0,00	6,25
19	lot_and_concession_number	2		0,80	0	NONE	0,00	0,00	NONE	N/A	N/A	0,80
20	section	2	TTM	115,76	3	TTM	0,00	114,97	NONE	N/A	N/A	115,76
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	40L0RU?L?1L7J0	40,67	3	4 0LORU?L?DLZJ0	2,61	28,42	4010 rue Belzile	28,57	28,57	43,28
28	mailing_country_unit	1	CANRLDA	6,69	1	('CANADA', 2)	0,17	0,00	Canada	66,67	100,00	6,86
29	mailing_province_unit	1	BPDHR	65,29	3	('BC', 4)	0,17	60,31	Québec	0,00	0,00	65,46
30	mailing_city_unit	1	YLHAISKL	38,27	3	('RIMOUSKI', 4)	0,43	30,77	Rimouski	50,00	100,00	38,70
31	mailing_postal_code_unit	3	L0L0Z3	4,56	3	L0L0Z3	0,00	0,00	66Z 3H0	0,00	0,00	4,56
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				483,00			54,20	386,56		61,57	66,58	537,2

Élections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
~~Chloé~~ Batman	Chloé	

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
2001 / 04 / 23	Masculin ☐ Féminin ☑ Genre X ☐	30 402 612		Français ☑ Anglais ☐

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue: 32 rue Lebel

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
	Québec	Québec	G0S 4S7

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : ☑

Adresse postale:

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale: 32 rue Lebel

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal
Québec	Québec	Canada	G0T 4S7

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : ☐

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

x Chloé Batman — *Signature de l'électeur des Forces canadiennes*

2024-04-24 — *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription: _____

Nom de la circinscription: _____

Signature du fonctionnaire électoral de l'unité: _____ Date: A A A A / M M / J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	TDDOB?MAN	81,83	1	('THOMAS', 5)	0,34	76,36	Batman	66,67	16,67	82,17
2	given_name	1	CWT	2,51	1	('COT', 1)	18,11	0,00	Chloé	20,00	20,00	20,62
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	2001	3,42	2	2001	0,00	0,00	2001	100,00	100,00	3,42
5	month_of_birth	0	4	1,38	2	4	0,00	0,00	04	100,00	100,00	1,38
6	day_of_birth	0	23	1,44	2	23	0,00	0,00	23	100,00	100,00	1,44
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	30402612	5,78	0	30402612	0,00	0,00	30 402 612	100,00	100,00	5,78
11	rank	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	32UUUW	30,81	3	32 UUUW	0,00	25,44	32 rue Lebel	30,00	30,00	30,81
15	apt./unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	WRBBC	46,19	1	('WHITBY', 4)	0,37	42,72	Québec	33,33	33,33	46,56
17	province	1	OUIWL	4,37	1	('NL', 4)	0,13	0,00	Québec	16,67	16,67	4,50
18	postal_code	3	B0R4	3,28	0	B0R4	0,00	0,00	G0J 4S7	33,33	33,33	3,28
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	?RVPLEBEL	25,02	3	2 RVPLEBEL	0,00	15,76	32 rue Lebel	70,00	80,00	25,02
28	mailing_country_unit	1	CANADA	5,68	1	('CANADA', 0)	0,14	0,00	Canada	100,00	100,00	5,82
29	mailing_province_unit	1	GLLIBTL	6,10	3	('ALBERTA', 5)	0,18	0,00	Québec	0,00	0,00	6,28
30	mailing_city_unit	1	GUTBEL	5,48	3	('SURREY', 4)	0,35	0,00	Québec	0,00	0,00	5,83
31	mailing_postal_code_unit	3	G8T4S7	4,70	3	G8T4S7	0,00	0,00	G0J 4S7	66,67	66,67	4,70
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				227,99			19,62	160,28		71,16	69,42	247,61

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Lemieux	Jérôme	

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1987 05 08	Male ☑ Female ☐ Gender X ☐	752808154	OS	English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street: 1050 avenue Bisson

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
	Rimouski	Québec	G4W 1H0

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)
—	—

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☑

Mailing address:

City, town, village or municipality	Province/territory/state	Country	Postal Code

Unit mailing address

Current mailing address: 1010 rue St-Simon

City, town, village or municipality	Province/territory/state	Country	Postal code
Bonaventure	Québec	Canada	320 1H0

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☐

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X Jérôme Lemieux — *Signature of Canadian Forces elector*

2024-02-21. — *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code: ____

ED name: ____

Unit election officer signature: ____ Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	L?DWLOXL?	34,34	1	('DION', 6)	0,34	26,44	Lemieux	28,57	0,00	34,68
2	given_name	1	LE?DQ?	66,50	1	('LEDIA', 3)	31,71	61,45	Jérôme	16,67	16,67	98,21
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	119?	23,44	2		0,00	20,80	1987	25,00	N/A	23,44
5	month_of_birth	0	5	0,68	2	5	0,00	0,00	5	100,00	100,00	0,68
6	day_of_birth	0	1779	31,21	2	IMPOSSIBLE VALUE	0,00	29,09	8	0,00	0,00	31,21
7	gender_male	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
8	gender_female	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	?2808154	59,67	0	?2808154	0,00	53,78	752808154	77,78	77,78	59,67
11	rank	2	?TT	49,29	0	?TT	0,00	48,24	5	0,00	0,00	49,29
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	LK?TL?1CV?LW?YYT	232,62	3	1 K?TL?LCV?LW?YYT	0,85	219,89	1050 avenue Bisson	0,00	6,25	233,47
15	apt,/unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	?OLWKID	45,87	1	('COLWOOD', 3)	0,42	39,76	Rimouski	25,00	0,00	46,29
17	province	1	??R	65,36	1	('AB', 3)	0,17	62,82	Québec	0,00	0,00	65,53
18	postal_code	3	R4I1L5L	5,44	0	R4I1L5L	0,00	0,00	G4W 1H0	16,67	16,67	5,44
19	lot_and_concession_number	2	?	10,97	0	?	0,00	10,19	NONE	N/A	N/A	10,97
20	section	2	?	21,12	3	?	0,00	20,33	NONE	N/A	N/A	21,12
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2	L0L?1WDV5TWGJMCYN	27,31	3	1010 1WDVSTWGJMCYN	0,88	10,15	1010 rue St-Simon	12,50	40,00	28,19
28	mailing_country_unit	1	?N?DQ	33,21	1	('INDE', 3)	0,21	28,93	Canada	33,33	0,00	33,42
29	mailing_province_unit	1	?PTTPC	54,62	3	?PTTPC	0,00	15,96	Québec	16,67	16,67	54,62
30	mailing_city_unit	1	?L?NTD?	81,88	3	?L?NTD?	0,00	75,85	Bonaventure	0,00	0,00	81,88
31	mailing_postal_code_unit	3	L3Z0L4	4,74	3	L3Z0L4	0,00	0,00	3Z0 1H0	50,00	50,00	4,74
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				848,27			34,58	723,68		50,09	48,87	882,85

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
fournier-charest	Vanessa	

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1994 05 23	Male ☐ Female ☒ Gender X ☐	038 846 211		English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street: 62 rue de tremble

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
	New Town	Manitoba	J4A 2B0

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☐

Mailing address: 42 chemin de lagrere

City, town, village or municipality	Province/territory/state	Country	Postal Code
matane	Quebec	Canada	G4A 1G2

Unit mailing address

Current mailing address

City, town, village or municipality	Province/territory/state	Country	Postal code

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☒

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X [signature] 23 01 04

Signature of Canadian Forces elector *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code

ED name

Unit election officer signature _____ Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	IJLLIRLDPVFTCHL?TCT	44,56	1	('LIRETTE', 14)	0,42	28,12	Fournier-Charest	18,75	0,00	44,98
2	given_name	1	V?LDLHIX	32,53	1	('VIDHI', 3)	40,64	25,87	Vanessa	14,29	14,29	73,17
3	middle_name	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
4	year_of_birth	0	1994	2,69	2	1994	0,00	0,00	1994	100,00	100,00	2,69
5	month_of_birth	0	544	49,39	2	IMPOSSIBLE VALUE	0,00	47,43	5	33,33	N/A	49,39
6	day_of_birth	0	63	7,73	2	IMPOSSIBLE VALUE	0,00	6,45	23	50,00	N/A	7,73
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	2887464211	26,05	0	2887464211	0,00	18,93	038846211	55,56	55,56	26,05
11	rank	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	62RULCX??J1YL??	106,85	3	6 2RULCX??JYYL??	0,99	93,67	62 rue de Tremble	28,57	28,57	107,84
15	apt,/unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	UCVUTYCLLL?	22,16	1	('UCLUELET', 7)	0,51	12,44	Newtown	0,00	0,00	22,67
17	province	1	?IGQDCL?Q	64,02	1	('QC', 7)	0,27	56,19	Manitoba	0,00	0,00	64,29
18	postal_code	3	M2M	2,41	0	M2M	0,00	0,00	J4A 2B0	0,00	0,00	2,41
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
22	mailing_address	2	LL?I?5C8JQKYN??JLCLCJPIC	69,34	3	612 I?SCIJQKYN??JLCLCJPIC	1,87	46,77	42 chemin de la grève	0,00	0,00	71,21
23	mailing_country	1	LCCKNDA	35,74	1	('CANADA', 4)	0,22	29,63	Canada	33,33	100,00	35,96
24	mailing_province	1	CIL?T?	70,41	3	('ALBERTA', 5)	0,26	89,64	Québec	0,00	0,00	70,67
25	mailing_city	1	LL?N	27,72	3	('ALMA', 3)	0,38	0,00	Matane	0,00	0,00	28,10
26	mailing_postal_code	3	L6J4L0L	5,36	3	L6J4L0L	0,00	0,00	G2A 102	16,67	16,67	5,36
27	mailing_address_unit	2		0,01	3	NONE	0,00	0,00	NONE	N/A	N/A	0,01
28	mailing_country_unit	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
29	mailing_province_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
30	mailing_city_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
31	mailing_postal_code_unit	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
32	personal_information	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				566,97			45,56	455,14		50,02	53,10	612,53

Elections Canada

Application for Registration and Special Ballot

For members of the Canadian Forces
(See section 221.2 of the *Canada Elections Act*)

EC 78014
(05/2019)

Full name and contact information

Family name	Given name	Middle name(s)
Fournier-Charest	Vanessa	Robin

Date of birth (yyyy-mm-dd)	Gender	Service number (mandatory)	Rank	Language
1994 / 02 / 23	Male ☐ Female ☒ Gender X ☐	038 441 462		English ☑ French ☐

Canadian address of ordinary residence (this determines the federal electoral district your vote will be counted in)

Number and street: 255 rue de la question

Apt./Unit	City, town, village or municipality	Province/territory	Postal code
	Saint-Clinclin	Quebec	

Lot and concession numbers (if applicable)	Section, township, range, meridian (if applicable)

Mailing address (where the voter information card is sent)

If same as address of ordinary residence, check here: ☒

Mailing address

City, town, village or municipality	Province/territory/state	Country	Postal Code

Unit mailing address

Current mailing address

City, town, village or municipality	Province/territory/state	Country	Postal code

Declaration

If you **do not** want your personal information to be added to the National Register of Electors, check here: ☐

I declare that:

- I am a Canadian citizen and will be at least 18 years old on election day.
- I have not already voted in this election, and I understand that I can only vote once.
- All of the statements on this form are true.

X [signature] 2021-02-21

Signature of Canadian Forces elector *Date*

If you make a false declaration, you could be fined and/or imprisoned.

Office use only

5-digit ED code

ED name

Unit election officer signature

Date: Y Y Y Y M M D D

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	FOURNIEMRTCHCLREST	16,12	1	('FOURNIER-CHAREST', 4)	0,40	0,00	Fournier-Charest	75,00	100,00	16,52
2	given_name	1	VCL	2,40	1	('VAL', 1)	18,97	0,00	Vanessa	14,29	28,57	21,37
3	middle_name	1	?OLGTN	12,56	1	('OLGUN', 2)	32,29	7,41	Robin	40,00	20,00	44,85
4	year_of_birth	0	1994	3,01	2	1994	0,00	0,00	1994	100,00	100,00	3,01
5	month_of_birth	0	22	1,34	2	IMPOSSIBLE VALUE	0,00	0,00	2	50,00	N/A	1,34
6	day_of_birth	0	23	1,35	2	23	0,00	0,00	23	100,00	100,00	1,35
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	038441462	7,11	0	038441462	0,00	0,00	038441462	100,00	100,00	7,11
11	rank	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
12	language_EN	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_FR	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	255RJCDCLA	9,15	3	2 55RJCDCLA	0,00	0,00	255 rue de la question	38,89	38,89	9,15
15	apt,/unit	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
16	city	1	JNTMCJCLNN	8,64	1	('MONCTON', 7)	0,44	0,00	Saint-clinclin	21,43	7,14	9,08
17	province	1	JCCLE	26,28	1	('BC', 4)	0,18	21,88	Québec	0,00	0,00	26,46
18	postal_code	3		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2		0,01	3	NONE	0,00	0,00	NONE	N/A	N/A	0,01
28	mailing_country_unit	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
29	mailing_province_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
30	mailing_city_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
31	mailing_postal_code_unit	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				87,97			52,28	29,29		74,42	76,15	140,25

Élections Canada

Demande d'inscription et de bulletin de vote spécial

À l'intention des membres des Forces canadiennes
(Voir les articles 221.2 de la *Loi électorale du Canada*)

EC 78014-1
(05/2019)

Nom et coordonnées

Nom de famille	Prénom	Autre(s) prénom(s)
Fournier-Charest	Vanessa	Batman

Date de naissance (aaaa-mm-jj)	Genre	Numéro matricule (obligatoire)	Grade	Langue
1994 02 23	Masculin ☐ Féminin ☒ Genre X ☐	025 810 592		Français ☑ Anglais ☐

Adresse de résidence habituelle au Canada (détermine la circonscription où votre vote sera compté)

Numéro et rue: 2 du centre-ville

App./Unité	Ville, village ou municipalité	Province ou territoire	Code postal
2	Oublie-Ville	Ontario	K4Y 7B1

Numéros de lot et de concession (le cas échéant)	Section, canton, rang et méridien (le cas échéant)

Adresse postale (où la carte d'information de l'électeur est envoyée)

Si elle est identique à votre adresse de résidence habituelle, cochez ici : ☒

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Adresse postale de l'unité

Adresse postale

Ville, village ou municipalité	Province, territoire ou état	Pays	Code postal

Déclaration

Si vous **ne** voulez **pas** que vos renseignements personnels soient ajoutés au Registre national des électeurs, cochez ici : ☐

J'atteste ce qui suit :

- Je suis citoyen canadien et j'aurai au moins 18 ans le jour de l'élection.
- Je n'ai pas déjà voté à cette élection, et je comprends que je peux voter qu'une seule fois.
- Toutes les déclarations faites sur ce formulaire sont vraies.

X Vanessa Fournier-Charest — *Signature de l'électeur des Forces canadiennes*

2024-04-10 — *Date*

Si vous faites une fausse déclaration, vous êtes passible d'une amende ou d'une peine d'emprisonnement ou des deux.

Réservé au bureau

Code à cinq chiffres de la circonscription: _____

Nom de la circinscription: _____

Signature du fonctionnaire électoral de l'unité: _____ Date: A A A A M M J J

Case ID	Contenu	Type de caractères (Chiffre: 0, lettres: 1, Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)	Texte reconnu	Temps requis reconnaissance (s)	Type de correction (0: Aucune, 1: Dictionnaire, 2: Date, 3: Adresse	Correction suggérée	Temps requis correction (s)	Temps fenêtrage (s)	Valeur réelle	Exactitude sans correction (%)	Exactitude avec correction (%)	Temps total (s)
1	family_name	1	L?CULLLCYMRLLICLIREFT	31,91	1	('FOURNIER-CHAREST', 14)	0,20	12,84	Fournier-Charest	12,50	100,00	32,11
2	given_name	1	LLUUILTSU	28,73	1	('LAURILOU', 4)	44,19	20,81	Vanessa	14,29	14,29	72,92
3	middle_name	1	TTTILT?YCN	66,61	1	('STILIYAN', 5)	49,15	59,96	Batman	33,33	33,33	115,76
4	year_of_birth	0	1194	29,36	2	1994	0,00	27,04	1994	75,00	100,00	29,36
5	month_of_birth	0	2	1,45	2	2	0,00	0,00	2	100,00	100,00	1,45
6	day_of_birth	0	23	1,59	2	23	0,00	0,00	23	100,00	100,00	1,59
7	gender_male	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
8	gender_female	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
9	gender_X	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
10	service_number	0	025 816 1851	8,66	0	025 816 1851	0,00	0,00	025 810 592	55,56	55,56	8,66
11	rank	2		0,01	0	NONE	0,00	0,00	NONE	N/A	N/A	0,01
12	language_FR	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
13	language_EN	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
14	number_and_street	2	2LLLKW	12,11	3	2 LLLKW	0,00	6,17	2 du centre-ville	6,67	6,67	12,11
15	apt,/unit	2	1?	9,31	3	1?	0,00	7,54	2	0,00	0,00	9,31
16	city	1	CCQBLBMU	7,78	1	('CHAPLEAU', 5)	0,21	0,00	oublie-ville	0,00	0,00	7,99
17	province	1	CRLMLCTVKC	34,79	1	('ALBERTA', 8)	0,01	25,91	Ontario	0,00	0,00	34,80
18	postal_code	3	T1I6L	3,89	0	T1I6L	0,00	0,00	G4W 7B1	0,00	0,00	3,89
19	lot_and_concession_number	2		0,00	0	NONE	0,00	0,00	NONE	N/A	N/A	0,00
20	section	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
21	same_ordinary_address	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
22	mailing_address	2		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
23	mailing_country	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
24	mailing_province	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
25	mailing_city	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
26	mailing_postal_code	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
27	mailing_address_unit	2		0,01	3	NONE	0,00	0,00	NONE	N/A	N/A	0,01
28	mailing_country_unit	1		0,00	1	NONE	0,00	0,00	NONE	N/A	N/A	0,00
29	mailing_province_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
30	mailing_city_unit	1		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
31	mailing_postal_code_unit	3		0,00	3	NONE	0,00	0,00	NONE	N/A	N/A	0,00
32	personal_information	4	[False]	0,00	0	[False]	0,00	0,00	False	100,00	100,00	0,00
33	signature	4	[True]	0,00	0	[True]	0,00	0,00	True	100,00	100,00	0,00
Total				236,21			93,76	160,27		59,87	65,49	329,97

ANNEXE V : PROGRAMME PRINCIPAL D'OCR

```
'''
Nom du programme : Programme principal d'OCR
Nom du fichier : ocr.py
Contenu du fichier : Programme principal pour le programme D'OCR, à appeler
à partir d'un main.py
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import pretraitement
import transform
import tflite
import cv2
import forms
import time
import utils
import posttraitement
import output
import numpy as np
from utils import bcolors
import prototypage

def get_streetNumber(imgROI, modelsOriginalChiffres, modelsDCTChiffres,
modelsDWTChiffres, modelsHoughChiffres, debug):
  '''
  Fonction tentant d'isoler le numéro de l'adresse en détectant le premier
contours
  Entrée: image brute, modèle de réseaux pour les chiffres (original, DCT,
DWT et Hough) et mode deboggage
  Sortie: Numéro détecté
  Par: Marc-Antoine Coulombe
  Dernière modification: 2024-03-31
  '''
  characs = []
  streetNumber = ""
  #Détecte le premier contours dans la case
  wordsPos = pretraitement.multipleContent_ROI(imgROI, debug = debug)
  if len(wordsPos) > 0:
    #Extrait la région de l'image avec le premier contour
    streetNumberROI =pretraitement.extract_contours(imgROI, wordsPos[0],
padding=False, debug = debug)
```

```
    charactersPos = pretraitement.contours_ROI(streetNumberROI, debug =
debug)
    #Pour chaque contours se trouvant dans la zone extraire, on isole le
caractère et le présente aux réseaux
    if len(charactersPos) > 0:
      for i in range(len(charactersPos)):
        imgCharacter = pretraitement.extract_contours(streetNumberROI,
charactersPos[i], padding=True, mask = True, debug = debug)
        imgCharacter, imgCharacterDCT, imgCharacterDWT, imgCharacterHough =
transform.generate_all_transforms(imgCharacter, debug = debug)
        charac, characPredValue = tflite.prediction_numbers(imgCharacter,
modelsOriginalChiffres, imgCharacterDCT, modelsDCTChiffres, imgCharacterDWT,
            modelsDWTChiffres, imgCharacterHough, modelsHoughChiffres)
        characs.append(charac)
      streetNumber = utils.chars_to_string(characs)
      if debug:
        print(f"Detected street value in case is: {streetNumber}")
    return streetNumber

def ocr_system(formID,formPath, templatePath, debugMode = False):
  '''
  Fonction principale du programme d'OCR
  Entrée: Code d'identification du formulaire, chemin d'accès du formulaire,
chemin d'accès vers le modèle et mode déboggage
  Sortie: Fichier .csv contenant les détections pour chacune des cases, les
temps de traitement et les corrections suggérées
  Par: Marc-Antoine Coulombe
  Dernière modification: 2024-03-31
  '''
  start_time = time.time()

  #Défini les chemins d'accès du répertoire utiliser pour les tests du
programme ainsi que pour enregistrer les données temporaires
  #et l'emplacement des modèles de réseaux de neurones
  pathProjetTest = '\\\\vmaw-dips-mus2d\\ec_avm_test_m.a.c\\Test'
  savePath = "D:\\Maitrise\\debug"
  modelPath = "C:\\Users\\coulomma\\OneDrive - SPAC-
PSPC\\Maitrise\\Programmation\\Reseaux neurones\\Reseaux\\Lite"
  #Chien de garde pour observer le répertoire en cas de nouveaux
formulaires, n'est pas implanté pour le moment.
  #a = os.listdir(pathProjetTest)
  #utils.activate_directory_handler(pathProjetTest)
```

```
  #Génère un fichier temporaire selon le formulaire pour enregistrer les
données et charge le formulaire et son modèle
  numberOfCases = forms.get_form_case(formID, 0)[0]
  output.generate_temp_header(pathProjetTest)

  imgForm = cv2.imread(formPath)
  imgForm = cv2.cvtColor(imgForm, cv2.COLOR_BGR2GRAY)

  imgTemplate = cv2.imread(templatePath)
  imgTemplate = cv2.cvtColor(imgTemplate, cv2.COLOR_BGR2GRAY)

  loading_endtime = time.time()
  print(f"Loading form: {loading_endtime-start_time:.2f} seconds")

  #Redimensionne le formulaire pour conrrespondre au modèle et on corrige
l'alignement, l'inclinaison et on réalise la soustraction
  imgForm = pretraitement.img_resize_to_ref(imgForm, imgTemplate)

  imgFormAligned = pretraitement.align_forms(imgForm, imgTemplate, formID,
debug = debugMode, savePath=savePath)
  imgFormSubstracted = pretraitement.substract_forms(imgFormAligned,
imgTemplate, formID, debug = debugMode, savePath=savePath)
  pretraitement_endtime = time.time()
  print(f"Preparation for recognition: {pretraitement_endtime-
loading_endtime:.2f} seconds")

  #Charge les modèles de réseaux de neurones en mémoire
  modelsOriginalChiffres, modelsDCTChiffres, modelsDWTChiffres,
modelsHoughChiffres = tflite.load_all_numbers_models(modelPath)
  modelsChiffres = [modelsOriginalChiffres, modelsDCTChiffres,
modelsDWTChiffres, modelsHoughChiffres]
  modelsOriginalLettres, modelsDCTLettres, modelsDWTLettres,
modelsHoughLettres = tflite.load_all_letters_models(modelPath)
  modelsLettres = [modelsOriginalLettres, modelsDCTLettres,
modelsDWTLettres, modelsHoughLettres]
  modelsOriginalMixte, modelsDCTMixte, modelsDWTMixte, modelsHoughMixte =
tflite.load_all_mixte_models(modelPath)
  modelsMixte = [modelsOriginalMixte, modelsDCTMixte, modelsDWTMixte,
modelsHoughMixte]
  loadingModels_endtime = time.time()
```

```
  print(f"Loading models into memory: {loadingModels_endtime-
pretraitement_endtime:.2f} seconds")

  #Initialise deux variables qui permettront d'aider à la détection dans la
boucle pour l'adresse et la case "Same address"
  country = ""
  sameAddress = False

  #Boucle pour chacune des cases dans le formulaire
  for caseID in range(1, numberOfCases + 1):

    #Extrait la case du formulaire selon le numéro de case
    imgROI = pretraitement.extract_ROI(imgFormSubstracted, formID,
caseID=caseID, debug = debugMode)

    #Détermine le contenu attendu dans la case, le type de caractère, le
type de correction et si l'on peut ignorer ces cases
    recognitionCase_starttime = time.time()
    expectedContent = forms.get_form_case(formID, caseID)[0]
    charactersType = forms.get_form_case(formID, caseID)[3]
    correctionType = forms.get_form_case(formID, caseID)[4]
    skipIfSameAddress = forms.get_form_case(formID, caseID)[5]

    #Initialise les listes de caractères ainsi que la correction suggérée
    correction = "NONE"
    charac = []
    characs = []
    streetNumber = ""
    charactersPos = []

    #Si on ne peut ignorer les cases, extrait les lettres et effectue la
reconnaissance
    if not(sameAddress and skipIfSameAddress == 1):

      #Si la case est une case à cocher, on regarde si la moyenne des pixels
dépassent un seuil pour savoir si la case est cochée
      if(charactersType == 4):
        _, imgROIThresh = cv2.threshold(imgROI, 20, 255, cv2.THRESH_BINARY)
        if imgROIThresh.mean() > 3:
          OCRText = [True]
        else:
          OCRText = [False]
```

```
      #Si le type attendu est parmi les types prévus, continue l'exécution
      elif (0 <= charactersType < 4):
        #Détermine la position du texte dans la case et extrait la région
        contentPos = pretraitement.content_ROI(imgROI, debug = debugMode)
        imgROI = pretraitement.extract_ROI(imgFormAligned, formID, caseID,
debug = debugMode)
        imgROI = cv2.bitwise_not(pretraitement.extract_contours(imgROI,
contentPos, padding=False, debug = debugMode))

        #Dans le cas où la case est une adresse, tente d'extraire le premier
contour comme numéro
        if expectedContent == "number_and_street" or expectedContent ==
"mailing_address" or expectedContent == "mailing_address_unit":
          streetNumber = get_streetNumber(imgROI, modelsOriginalChiffres,
modelsDCTChiffres, modelsDWTChiffres, modelsHoughChiffres, debug =
debugMode)

        #Détermine tous les contours se trouvant dans la région de texte
        charactersPos = pretraitement.contours_ROI(imgROI, debug =
debugMode)
        #Si la région n'est pas vide, on passe à l'extraction des
caractéristiques
        if len(charactersPos) > 0:

          #Pour chacun des contours, extrait celui-ci de l'image et génère
ses transformées DCT, DWT et Hough
          for i in range(len(charactersPos)):
            imgCharacter = pretraitement.extract_contours(imgROI,
charactersPos[i], mask=True, padding=True, debug = debugMode)
            imgCharacter, imgCharacterDCT, imgCharacterDWT,
imgCharacterHough = transform.generate_all_transforms(imgCharacter, debug =
debugMode)

            #Selon le type de caractère, on présente aux réseaux appropriés
            #Type 0 = Chiffre
            #Type 1 = Lettres
            #Type 2 = Combinaison chiffres + Lettres (utilisation des
réseaux mixtes pour faire la distinction)
            #Type 3 = Alternance Lettres + Chiffres (Code postaux canadiens)
            if (charactersType == 0):
```

```
            charac, characPredValue =
tflite.prediction_numbers(imgCharacter, modelsOriginalChiffres,
imgCharacterDCT, modelsDCTChiffres, imgCharacterDWT,
              modelsDWTChiffres, imgCharacterHough, modelsHoughChiffres,
debug=False)

            #Si la prédiction est inférieur un seuil, pose l'hypothèse
qu'il peut y avoir plus d'un caractère dans le contour
            #Seuil a déterminer dans un développement futur du projet
            if characPredValue < 0.7:
              charac =
utils.extract_element(prototypage.separate_characters(imgROI,
charactersPos[i], charac, charactersType, modelsOriginalChiffres,
modelsDCTChiffres,
                modelsDWTChiffres, modelsHoughChiffres, debug =
debugMode), 0)

          elif (charactersType == 1):
            charac, characPredValue =
tflite.prediction_letters(imgCharacter, modelsOriginalLettres,
imgCharacterDCT, modelsDCTLettres, imgCharacterDWT,
              modelsDWTLettres, imgCharacterHough, modelsHoughLettres,
debug=False)

            #Si la prédiction est inférieur un seuil, pose l'hypothèse
qu'il peut y avoir plus d'un caractère dans le contour
            #Seuil a déterminer dans un développement futur du projet
            if characPredValue < 0.6:
              charac =
utils.extract_element(prototypage.separate_characters(imgROI,
charactersPos[i], charac, charactersType, modelsOriginalLettres,
modelsDCTLettres,
                modelsDWTLettres, modelsHoughLettres, debug = debugMode),
0)

          elif (charactersType == 2):
            predType, predTypeValue =
tflite.prediction_mixte(imgCharacter, modelsOriginalMixte, imgCharacterDCT,
modelsDCTMixte, imgCharacterDWT,
                modelsDWTMixte, imgCharacterHough, modelsHoughMixte,
debug=False)
```

```
            if predType == 0:
              charac, characPredValue =
tflite.prediction_numbers(imgCharacter, modelsOriginalChiffres,
imgCharacterDCT, modelsDCTChiffres, imgCharacterDWT,
                  modelsDWTChiffres, imgCharacterHough,
modelsHoughChiffres, debug=False)

              #Si la prédiction est inférieur un seuil, pose l'hypothèse
qu'il peut y avoir plus d'un caractère dans le contour
              if characPredValue < 0.7:
                charac =
utils.extract_element(prototypage.separate_characters(imgROI,
charactersPos[i], charac, predType, modelsOriginalChiffres,
modelsDCTChiffres,
                  modelsDWTChiffres, modelsHoughChiffres, debug =
debugMode), 0)

            elif predType == 1:
              charac, characPredValue =
tflite.prediction_letters(imgCharacter, modelsOriginalLettres,
imgCharacterDCT, modelsDCTLettres, imgCharacterDWT,
                  modelsDWTLettres, imgCharacterHough, modelsHoughLettres,
debug=False)

              #Si la prédiction est inférieur un seuil, pose l'hypothèse
qu'il peut y avoir plus d'un caractère dans le contour
              #Seuil a déterminer dans un développement futur du projet
              if characPredValue < 0.6:
                charac =
utils.extract_element(prototypage.separate_characters(imgROI,
charactersPos[i], charac, predType, modelsOriginalLettres, modelsDCTLettres,
                  modelsDWTLettres, modelsHoughLettres, debug =
debugMode), 0)

            else:
                print(bcolors.FAIL + "ERREUR DANS LA DÉTECTION DU TYPE
LORS DE LA PRÉDICTION MIXTE" + bcolors.ENDC)
                exit()

          elif (charactersType == 3):
            if len(charactersPos) == 6:
              if i % 2 != 0:
```

```
                charac, characPredValue =
tflite.prediction_numbers(imgCharacter, modelsOriginalChiffres,
imgCharacterDCT, modelsDCTChiffres, imgCharacterDWT,
                    modelsDWTChiffres, imgCharacterHough,
modelsHoughChiffres)
              else:
                charac, characPredValue =
tflite.prediction_letters(imgCharacter, modelsOriginalLettres,
imgCharacterDCT, modelsDCTLettres, imgCharacterDWT,
                    modelsDWTLettres, imgCharacterHough,
modelsHoughLettres)
            else:
              if i % 2 != 0:
                charac, characPredValue =
tflite.prediction_numbers(imgCharacter, modelsOriginalChiffres,
imgCharacterDCT, modelsDCTChiffres, imgCharacterDWT,
                    modelsDWTChiffres, imgCharacterHough,
modelsHoughChiffres)
              else:
                charac, characPredValue =
tflite.prediction_letters(imgCharacter, modelsOriginalLettres,
imgCharacterDCT, modelsDCTLettres, imgCharacterDWT,
                    modelsDWTLettres, imgCharacterHough,
modelsHoughLettres)
              #if characPredValue < 0.7:
              #  charac =
utils.extract_element(prototypage.separate_characters(imgROI,
charactersPos[i], charac, predType, modelsOriginalLettres, modelsDCTLettres,
              #    modelsDWTLettres, modelsHoughLettres, debug = False),
0)

          characs.append(charac)

      #S'il ne s'agit aucun des quatres types préprogrammés, on retourne un
message d'erreur.
      else:
        print(utils.bcolors.FAIL + "Erreur détectée: Action non-définie
main.py ligne 47" + utils.bcolors.ENDC)
        exit()

    recognitionCase_endtime = time.time()
```

```
    elapsedRecognition = "{:.2f}".format(recognitionCase_endtime-
recognitionCase_starttime)
    correctionCase_starttime = time.time()

    #Si le type n'est pas un booléen, converti les caractères en chaine de
caractères
    if (charactersType != 4):
      OCRText = utils.chars_to_string(characs)
    #Si la case n'est pas vide, on effectue un post-traitement selon le type
de correction
    if OCRText != "":
      correction = posttraitement.correction_by_case(OCRText,
correctionType, expectedContent, streetNumber, imgROI, formID, caseID,
country, modelsLettres, debug = debugMode)
    print(f"Detected value in case {caseID} is: {correction}")

    #Si la case était le pays, on conserve la données afin de comparer lors
du post-traitement
    #S'il s'agissait de la case "Same address", on conserve l'information
pour pouvoir potentiellement sauter des cases dans la suite du formulaire
    if "country" in expectedContent:
      country = correction[0]
    elif "same_ordinary_address" in expectedContent:
      sameAddress = OCRText[0]

    correctionCase_endtime = time.time()
    elapsedCorrection = "{:.2f}".format(correctionCase_endtime-
correctionCase_starttime)
    #Écrit les résultats dans un fichier temporaire (une ligne = une case)
    tempRow = [expectedContent, charactersType, OCRText, elapsedRecognition,
correctionType, correction, elapsedCorrection]
    output.export_to_temp(tempRow, pathProjetTest)

  #Écrit les résultats dans un fichier final (une ligne = un formulaire)
  output.generate_output_file(f"{formID}.csv", pathProjetTest, debug = True)
```

ANNEXE VI : PROGRAMME DE PRÉTRAITEMENT

```
'''
Nom du programme : Programme de prétraitement
Nom du fichier : pretraitement.py
Contenu du fichier : Fonctions pour le prétraitement pour du système D'OCR
(correction de l'alignement, filtrage, extraction des cases et segmentation)
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import cv2
import numpy as np
import forms


def img_resize_scale(img, scale):
    '''
    Fonction permettant de redimensionner une image
    Entrée: Image originale, facteur d'échelle
    Sortie: Image redimensionnée
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    width = int(img.shape[1] * scale)
    Height = int(img.shape[0] * scale)
    dim = (width, Height)
    resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
    return resized

def img_resize_to_ref(img, ref):
    '''
    Fonction permettant de redimensionner une image selon une autre image de
référence
    Entrée: Image originale, image de référence
    Sortie: Image redimensionnée
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    width = int(ref.shape[1])
    Height = int(ref.shape[0])
    dim = (width, Height)
    resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
```

```
    return resized

def img_resize_width(img, width):
    '''
    Fonction permettant de redimensionner une image selon une largeur
spécifiée
    Entrée: Image originale, image de référence
    Sortie: Image redimensionnée
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    Height = int(img.shape[0])
    dim = (width, Height)
    resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
    return resized

def align_forms(imgForm, imgTemplate, formID, maxFeatures = 5000,
keepPercent = 0.1, debug = False, savePath=None):
    '''
    Fonction permettant d'aligner deux formulaires à partir d'un modèle
    Entrée: Image du formulaire rempli, image du modèle du formaire, numéro
de formulaire, répertoire de sortie pour sauvegarder le résultat, le nombre
de points à détecter
            le pourcentage de points à conserver et un booléan si l'on
souhaite utiliser la fonction debuggage
    Sortie: Retourne l'image aligné avec le formulaire
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-17
    Référence: Rosebrock, A., Thanki, A., Paul, S., & Haase, J. (2020). OCR
with OpenCV, Tesseract, and Python - Intro to OCR.
    '''

    # use ORB to detect keypoints and extract (binary) local invariant
features
    orb = cv2.ORB_create(maxFeatures)
    (kpsA, descsA) = orb.detectAndCompute(imgForm, None)
    (kpsB, descsB) = orb.detectAndCompute(imgTemplate, None)

    # match the features
    method = cv2.DescriptorMatcher_BRUTEFORCE_HAMMING
    matcher = cv2.DescriptorMatcher_create(method)
    matches = matcher.match(descsA, descsB, None)
```

```
    # sort the matches by their distance (the smaller the distance, the
"more similar" the features are)
    matches = sorted(matches, key=lambda x:x.distance)

    # keep only the top matches
    keep = int(len(matches) * keepPercent)
    matches = matches[:keep]

    # allocate memory for the keypoints (x,y-coordinates) from the top
matches -- we'll use these coordinates to compute our homography matrix
    ptsA = np.zeros((len(matches), 2), dtype="float")
    ptsB = np.zeros((len(matches), 2), dtype="float")

    # loop over the top matches
    for (i, m) in enumerate(matches):
        # indicate that the two keypoints in the respective images map to
each other
        ptsA[i] = kpsA[m.queryIdx].pt
        ptsB[i] = kpsB[m.trainIdx].pt

    # compute the homography matrix between the two sets of matched points
    (H, mask) = cv2.findHomography(ptsA, ptsB, method=cv2.RANSAC,
ransacReprojThreshold = 10)

    # use the homography matrix to align the form
    (h, w) = imgTemplate.shape[:2]
    imgAlignedForm = cv2.warpPerspective(imgForm, H, (w, h))

    if debug:
        matchedVis = cv2.drawMatches(imgForm, kpsA, imgTemplate, kpsB,
matches, None)
        matchedVisResized = img_resize_scale(matchedVis, 0.4)
        cv2.imshow("Matches", matchedVisResized)

        imgOverlap = imgTemplate + imgAlignedForm
        imgOverlapResized = img_resize_scale(imgOverlap, 0.4)
        cv2.imshow("Form overlap", imgOverlapResized)

        imgAlignedFormResized = img_resize_scale(imgAlignedForm, 0.4)
        cv2.imshow("Form aligned", imgAlignedFormResized)
        cv2.waitKey(0)
```

```
        cv2.destroyAllWindows()

    if savePath is not None:
        matchedVis = cv2.drawMatches(imgForm, kpsA, imgTemplate, kpsB,
matches, None)
        imgOverlap = imgTemplate + imgAlignedForm
        cv2.imwrite(f"{savePath}\\{formID}_matches.jpg", matchedVis)
        cv2.imwrite(f"{savePath}\\{formID}_overlap.jpg", imgOverlap)
        cv2.imwrite(f"{savePath}\\{formID}_aligned.jpg", imgAlignedForm)

    # return the aligned image
    return imgAlignedForm

def substract_forms(imgFormAligned, imgTemplate, formID, debug = False,
savePath = None):
    '''
    Fonction permettant de soustraire le modèle au formulaire fourni afin de
ne conserver que l'écriture manuscrite
    Input: Le formulaire aligné au modèle, le modèle de formulaire.
    Output: Le formulaire contenant uniquement les informations manuscrites.
    Par: Marc-Antoine Coulombe
    Date: 2023-10-17
    '''
    kernel = np.ones((3,3), np.uint8)
    imgTemplate = cv2.erode(imgTemplate, kernel, iterations = 3)
    imgFormAlignedInv = cv2.bitwise_not(imgFormAligned)
    imgTemplateInv = cv2.bitwise_not(imgTemplate)

    imgFormSubstracted = cv2.subtract(imgFormAlignedInv, imgTemplateInv)

    imgFormSubstracted = cv2.medianBlur(imgFormSubstracted, 5)
    imgFormSubstracted = cv2.GaussianBlur(imgFormSubstracted, (13, 13), 0)

    if debug:
        imgFormAlignedResized = img_resize_scale(imgFormAligned, 0.4)
        cv2.imshow("Aligned form", imgFormAlignedResized)

        imgTemplateInvResized = img_resize_scale(imgTemplateInv, 0.4)
        cv2.imshow("Template inverted", imgTemplateInvResized)

        imgFormSubstractedResized = img_resize_scale(imgFormSubstracted,
0.4)
```

```
        cv2.imshow("Substracted form", imgFormSubstractedResized)
        cv2.waitKey(0)
        cv2.destroyAllWindows()

    if savePath is not None:
        cv2.imwrite(f"{savePath}\\{formID}_Formulaire_Soustrait.jpg",
imgFormSubstracted)

    return imgFormSubstracted

def extract_ROI(imgFormSubstracted, formID, caseID, debug = False):
    '''
    Fonction permettant d'extraire une case du formulaire
    Entrée: Formulaire soustrait, numéro de formulaire, numéro de case,
debug, chemin d'accès pour sauvegarde des images
    Sortie: Contours dans l'image triée de gauche à droite
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    caseCoord = forms.get_form_case(formID, caseID)
    imgROI = imgFormSubstracted[caseCoord[1][1]:caseCoord[2][1],
caseCoord[1][0]:caseCoord[2][0]]
    if debug:
        imgFormSubstractedResized = img_resize_scale(imgFormSubstracted,
0.4)
        cv2.imshow("Substracted form", imgFormSubstractedResized)

        imgROIResized = img_resize_scale(imgROI, 0.4)
        cv2.imshow("cropped", imgROIResized)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
    return imgROI

def overlapping_boxes(boxA, boxB):
    '''
    Fonction permettant de déterminer si deux rectangles s'intersectent
    Entrée: Rectangle A, Rectangle B
    Sortie: Booléen si intersection
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    return not (boxA[0]+boxA[2]-2 < boxB[0]
```

```
                or boxA[0] > boxB[0]+boxB[2]-2
                or boxA[1] > boxB[1]+boxB[3]-2
                or boxA[1]+boxA[3]-2 < boxB[1])

def fusion_overlapping_boxes(filteredBoxes, meanDeviation = 10):
    '''
    Fonction permettant de fusionner des contours qui se superposent
    Entrée: Liste de contours, nombre de pixels de décalage tolérer (centre-
à-centre)
    Sortie: Liste des contours avec les contours qui se superposent
fusionnés
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    topY = 0
    bottomY = 0
    leftX = 0
    rightX = 0
    for k in range(0, 2):
        for i in range(len(filteredBoxes)):
            for j in range(len(filteredBoxes)):
                if i != j:
                    box1 = [filteredBoxes[i][0], filteredBoxes[i][1],
filteredBoxes[i][2], filteredBoxes[i][3]]
                    box1[1] = 0
                    box2 = [filteredBoxes[j][0], filteredBoxes[j][1],
filteredBoxes[j][2], filteredBoxes[j][3]]
                    box2[1] = 0
                    if overlapping_boxes(box1, box2):
                        boxjAverage =
(filteredBoxes[j][0]+filteredBoxes[j][0]+filteredBoxes[j][2])/2
                        boxiAverage =
(filteredBoxes[i][0]+filteredBoxes[i][0]+filteredBoxes[i][2])/2
                        if (boxiAverage-meanDeviation <= boxjAverage
<=boxiAverage+meanDeviation):
                            if filteredBoxes[j][1] <= filteredBoxes[i][1]:
                                topY = filteredBoxes[j][1]
                            else:
                                topY = filteredBoxes[i][1]

                            if filteredBoxes[j][1] + filteredBoxes[j][3] <=
filteredBoxes[i][1] + filteredBoxes[i][3]:
```

```
                                bottomY = filteredBoxes[i][1] +
filteredBoxes[i][3]
                            else:
                                bottomY = filteredBoxes[j][1] +
filteredBoxes[j][3]

                            filteredBoxes[i][1] = topY
                            filteredBoxes[i][3] = bottomY - topY
                            filteredBoxes[i].append(filteredBoxes[j][4])
                            filteredBoxes[j] = [0, 0, 0, 0, 0]

    filteredBoxes = list(filter(lambda a: a != [0, 0, 0, 0, 0],
filteredBoxes))
    return filteredBoxes

def filtering_boxes(boxes, checkOverlapping=True, minimumArea = 30):
    '''
    Fonction de retirer les contours de petites dimensions et retirer les
contours nulles.
    Entrée: Liste des rectangles, aire minimale tolérée
    Sortie: Liste des rectangles filtrée
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    for i in range(len(boxes)):
        approximatedArea = boxes[i][2]*boxes[i][3]
        if(approximatedArea > minimumArea):
            pass
        else:
            boxes[i] = [0, 0, 0, 0, 0]
    filteredBoxes = list(filter(lambda a: a != [0, 0, 0, 0, 0], boxes))

    return filteredBoxes

def sort_contours(cnts, method="left-to-right"):
    '''
    Fonction permettant de trier une liste de contours de gauche à droite
    Entrée: Liste des contours
    Sortie: Contours dans l'image triée de gauche à droite
    Par: Marc-Antoine Coulombe
    Date: 2015-04-20
```

```
    Référence: https://pyimagesearch.com/2015/04/20/sorting-contours-using-
python-and-opencv/
    '''
    if cnts:
        # initialize the reverse flag and sort index
        reverse = False
        i = 0
        # handle if we need to sort in reverse
        if method == "right-to-left" or method == "bottom-to-top":
            reverse = True
        # handle if we are sorting against the y-coordinate rather than
        # the x-coordinate of the bounding box
        if method == "top-to-bottom" or method == "bottom-to-top":
            i = 1
        # construct the list of bounding boxes and sort them from top to
        # bottom
        boundingBoxes = [cv2.boundingRect(c) for c in cnts]
        (cnts, boundingBoxes) = zip(*sorted(zip(cnts, boundingBoxes),
            key=lambda b:b[1][i], reverse=reverse))
        # return the list of sorted contours and bounding boxes
        return (cnts, boundingBoxes)
    else:
        return ([], [])

def content_ROI(imgROI, debug = False):
    '''
    Fonction permettant de détecter le plus grand contour dans une case afin
d'en extraire le bloc de texte
    Entrée: Image de la case, numéro de formulaire, numéro de case,
        mode debug et chemin du répertoire si l'on souhaite sauvegarder les
images.
    Sortie: Image du bloc de texte se trouvant dans la case
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    imgROI = cv2.adaptiveThreshold(imgROI, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY, 15, -10)
    contours, _ = cv2.findContours(imgROI, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    if len(contours)>0:
        _, imgROI = cv2.threshold(imgROI, 0, 255,
cv2.THRESH_BINARY+cv2.THRESH_OTSU)
```

```
        kernel = np.ones((5,5),np.uint8)
        imgROI = cv2.dilate(imgROI, kernel, iterations = 17)
        contours, _ = cv2.findContours(imgROI, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
        contours, _ = sort_contours(contours)
        contoursROI = imgROI.copy()
    #loop over the contours
    boxes = []
    for c in contours:
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.02*peri, True)
        (x, y, w, h) = cv2.boundingRect(c)
        if x>0:
            x-=1
            w+=2
        if y>0:
            y-=1
            h+=2
        boxes.append([x, y, w, h])

    boxesfiltered = filtering_boxes(boxes, minimumArea = 200)
    minX = 9999
    maxX = 0
    minY = 9999
    maxY = 0
    maxW = 0
    maxH = 0

    if(len(boxesfiltered) > 0):
        for j in range (0, 2):
            for i in range(len(boxesfiltered)):
                if boxesfiltered[i][0] < minX:
                    minX = boxesfiltered[i][0]
                if (boxesfiltered[i][0]+boxesfiltered[i][2]) > maxX:
                    maxX = boxesfiltered[i][0]+boxesfiltered[i][2]
                if boxesfiltered[i][1] < minY:
                    minY = boxesfiltered[i][1]
                if (boxesfiltered[i][1]+boxesfiltered[i][3]) > maxY:
                    maxY = boxesfiltered[i][1]+boxesfiltered[i][3]
        boxesfiltered = [[minX, minY, maxX-minX, maxY-minY]]
```

```
    boxesfiltered = list(filter(lambda a: a != [0, 0, 0, 0, 0],
boxesfiltered))

    boxesfiltered = [val for sublist in boxesfiltered for val in sublist]

    if(len(boxesfiltered)>0):
        cv2.rectangle(contoursROI, (boxesfiltered[0], boxesfiltered[1]),
(boxesfiltered[0]+boxesfiltered[2], boxesfiltered[1]+boxesfiltered[3]),
(255, 255, 255), 2)

    if debug:
        imgROIResized = img_resize_scale(imgROI, 0.4)
        cv2.imshow("ROI", imgROIResized)
        if(len(boxesfiltered)>0):
            contoursROIResized = img_resize_scale(contoursROI, 1)
            cv2.imshow("Contours", contoursROIResized)
        cv2.waitKey(0)
        cv2.destroyAllWindows()

    return boxesfiltered

def multipleContent_ROI(imgROI, debug = False):
    '''
    Fonction permettant de détecter le plus grand contour dans une case afin
d'en extraire le bloc de texte
    Entrée: Image de la case, numéro de formulaire, numéro de case,
        mode debug et chemin du répertoire si l'on souhaite sauvegarder les
images.
    Sortie: Image du bloc de texte se trouvant dans la case
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''

    try:
        _, imgROI = cv2.threshold(imgROI, 0, 255,
cv2.THRESH_BINARY+cv2.THRESH_OTSU)
        kernel = np.ones((3,3),np.uint8)
        imgROI = cv2.dilate(imgROI, kernel, iterations = 3)
        #kernel = np.ones((4,4),np.uint8)
        #imgROI = cv2.erode(imgROI, kernel, iterations = 1)
        #kernel = np.ones((3,3),np.uint8)
        #imgROI = cv2.erode(imgROI, kernel, iterations = 2)
```

```
        contours, _ = cv2.findContours(imgROI, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
        contours, _ = sort_contours(contours)
        contoursROI = imgROI.copy()
        #loop over the contours
        boxes = []
        for c in contours:
            peri = cv2.arcLength(c, True)
            approx = cv2.approxPolyDP(c, 0.02*peri, True)
            (x, y, w, h) = cv2.boundingRect(c)
            boxes.append([x-1, y-1, w+2, h+2])

        boxesfiltered = filtering_boxes(boxes, 200)

        boxesfiltered = list(filter(lambda a: a != [0, 0, 0, 0],
boxesfiltered))
        for box in boxesfiltered:
            cv2.rectangle(contoursROI, (box[0], box[1]), (box[0]+box[2],
box[1]+box[3]), (255, 255, 255), 2)

        if debug:
            imgROIResized = img_resize_scale(imgROI, 0.4)
            cv2.imshow("ROI", imgROIResized)

            contoursROIResized = img_resize_scale(contoursROI, 1)
            cv2.imshow("Contours", contoursROIResized)
            cv2.waitKey(0)
            cv2.destroyAllWindows()
        return boxesfiltered
    except:
        return []

def contours_ROI(imgROI, debug = False, blocksize = 5):
    '''
    Fonction permettant de détecter les contours dans l'image afin d'en
extraire les caractères
    Entrée: Image de la case, numéro de formulaire, numéro de case,
        mode debug et chemin du répertoire si l'on souhaite sauvegarder les
images.
    Sortie: Image du caractère segmenté et centré
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
```

```
    '''

    try:
        imgROIThresh = cv2.adaptiveThreshold(imgROI, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY, blocksize, -10)
        contours, _ = cv2.findContours(imgROI, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
        if len(contours)>0:
            _, imgROIThresh = cv2.threshold(imgROI, 0, 255,
cv2.THRESH_BINARY+cv2.THRESH_OTSU)
            contours, _ = cv2.findContours(imgROIThresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
            contours, _ = sort_contours(contours)
        contoursROI = imgROIThresh.copy()
        #loop over the contours
        boxes = []
        for c in contours:
            peri = cv2.arcLength(c, True)
            approx = cv2.approxPolyDP(c, 0.02*peri, True)
            (x, y, w, h) = cv2.boundingRect(c)
            boxes.append([x, y, w, h, [c]])

        boxesfiltered = filtering_boxes(boxes, minimumArea=20)
        boxesfiltered = fusion_overlapping_boxes(boxesfiltered)

        for box in boxesfiltered:
            cv2.rectangle(contoursROI, (box[0], box[1]), (box[0]+box[2],
box[1]+box[3]), (255, 255, 255), 2)

        if debug:
            imgROIThreshResized = img_resize_scale(imgROIThresh, 0.4)
            cv2.imshow("ROI", imgROIThreshResized)

            contoursROIResized = img_resize_scale(contoursROI, 0.8)
            cv2.imshow("Contours", contoursROIResized)
            cv2.waitKey(0)
            cv2.destroyAllWindows()
        return boxesfiltered
    except:
        return []
```

```
def extract_contours(imgForm, box, mask=False, padding = False, debug =
False):
    '''
    Fonction permettant d'extraire une lettre du formulaire à partir de ses
coordonnées.
    Entrée: Image du formulaire aligné, coordonnées du caractère, numéro de
formulaire, numéro de case et position de la lettre,
        Padding pour centrer l'image, mode debug et chemin du répertoire si
l'on souhaite sauvegarder les images.
    Sortie: Image du caractère segmenté et centré
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    try:
        if mask:
            imgMask = np.zeros(imgForm.shape, dtype=imgForm.dtype)
            if len(box) == 5:
                cv2.drawContours(imgMask, box[4], -1, (255),
thickness=cv2.FILLED)
            else:
                for j in range(4, len(box)):
                    for i in range(0, len(box[j])):
                        cv2.drawContours(imgMask, [box[j][i]], -1, (255),
thickness=cv2.FILLED)
            masked = cv2.bitwise_and(imgForm, imgForm, mask=imgMask)
            imgCharac = masked[box[1]:(box[1]+box[3]),
box[0]:(box[0]+box[2])]
        else:
            imgCharac = imgForm[box[1]:(box[1]+box[3]),
box[0]:(box[0]+box[2])]

        if padding:
            width = imgCharac.shape[0]
            widthPadding = 0
            height = imgCharac.shape[1]
            heightPadding = 0

            if (height < width):
                heightPadding = int((width - height)/2)
                widthPadding = 0
            elif (width < height):
                heightPadding = 0
```

```
                widthPadding = int((height - width)/2)
            else:
                pass
            imgCharac = cv2.copyMakeBorder(imgCharac, widthPadding,
widthPadding, heightPadding, heightPadding, cv2.BORDER_CONSTANT, (0))
            imgCharacCentered = cv2.resize(imgCharac, (24, 24))
            imgCharacCentered = cv2.copyMakeBorder(imgCharacCentered, 2, 2,
2, 2, cv2.BORDER_CONSTANT, (0))
        else:
            imgCharacCentered = cv2.copyMakeBorder(imgCharac, 0, 0, 0, 0,
cv2.BORDER_CONSTANT, (0))
        if debug:
            if mask:
                imgMaskResized = img_resize_scale(imgMask, 1.2)
                cv2.imshow("imgMaskResized", imgMaskResized)
            else:
                imgFormResized = img_resize_scale(imgForm, 1.2)
                cv2.imshow("imgFormResized", imgFormResized)
            cv2.imshow("Character Centered", imgCharacCentered)
            cv2.waitKey(0)
            cv2.destroyAllWindows()

        return imgCharacCentered
    except:
        return 0

def threshold_character(imgCharacter, valmin=0, valMax = 255, debug =
False):
    '''
    Fonction permet de seuiller l'image d'un caractère
    Entrée: image du caractère segmenté, valeur de pixel minimale, valeur de
pixel maximiale (par défaut 8 bits) et si l'on souhaite utiliser le debug
    Sortie: Image du caractère seuillé entre aux bornes désirées
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    _, imgCharacter = cv2.threshold(imgCharacter, valmin, valMax,
cv2.THRESH_BINARY)
    if debug:
        cv2.imshow("imgCharacter", imgCharacter)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
```

```
return imgCharacter
```

ANNEXE VII : PROGRAMME DES INFORMATIONS SPÉCIFIQUES AUX FORMULAIRES

```
'''
Nom du programme : Programme des informations spécifiques aux formulaires
Nom du fichier : forms.py
Contenu du fichier : Coordonnées, contenu et type de corrections des cases
selon le modèle de formulaire
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

from utils import bcolors

def form_78014(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (33, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (103, 305), (597, 372), 1, 1, 0)
        case 2:
            return ("given_name", (602, 305), (1097, 372), 1, 1, 0)
        case 3:
            return ("middle_name", (1103, 305), (1594, 372), 1, 1, 0)
        case 4:
            return ("year_of_birth", (127, 430), (272, 497), 0, 2, 0)
        case 5:
            return ("month_of_birth", (302, 430), (422, 497), 0, 2, 0)
```

```
        case 6:
            return ("day_of_birth", (452, 430), (572, 497), 0, 2, 0)
        case 7:
            return ("gender_male", (754, 454), (770, 470), 4, 0, 0)
        case 8:
            return ("gender_female", (754, 479), (770, 495), 4, 0, 0)
        case 9:
            return ("gender_X", (754, 504), (770, 520), 4, 0, 0)
        case 10:
            return ("service_number", (828, 430), (1172, 497), 0, 0, 0)
        case 11:
            return ("rank", (1228, 430), (1347, 497), 2, 0, 0)
        case 12:
            return ("language_EN", (1529, 454), (1545, 470), 4, 0, 0)
        case 13:
            return ("language_FR", (1529, 479), (1545, 495), 4, 0, 0)
        case 14:
            return ("number_and_street", (103, 625), (1594, 697), 2, 3, 0)
        case 15:
            return ("apt./unit", (103, 730), (297, 797), 2, 3, 0)
        case 16:
            return ("city", (303, 730), (847, 797), 1, 1, 0)
        case 17:
            return ("province", (852, 730), (1347, 797), 1, 1, 0)
        case 18:
            return ("postal_code", (1353, 730), (1594, 796), 3, 0, 0)
        case 19:
            return ("lot_and_concession_number", (103, 830), (847, 897), 2,
0, 0)
        case 20:
            return ("section", (852, 830), (1594, 897), 2, 3, 0)
        case 21:
            return ("same_ordinary_address", (732, 945), (755, 968), 4, 0,
0)
        case 22:
            return ("mailing_address", (103, 1005), (1594, 1072), 2, 3, 1,
1)
        case 23:
            return ("mailing_country", (1077, 1105), (1372, 1172), 1, 1, 1)
        case 24:
            return ("mailing_province", (652, 1105), (1072, 1172), 1, 3, 1)
        case 25:
```

```
            return ("mailing_city", (103, 1105), (647, 1172), 1, 3, 1)
        case 26:
            return ("mailing_postal_code", (1377, 1105), (1594, 1171), 3, 3,
1)
        case 27:
            return ("mailing_address_unit", (103, 1255), (1594, 1322), 2, 3,
0)
        case 28:
            return ("mailing_country_unit", (1077, 1355), (1372, 1422), 1,
1, 0)
        case 29:
            return ("mailing_province_unit", (652, 1355), (1072, 1422), 1,
3, 0)
        case 30:
            return ("mailing_city_unit", (103, 1355), (647, 1422), 1, 3, 0)
        case 31:
            return ("mailing_postal_code_unit", (1377, 1355), (1594, 1421),
3, 3, 0)
        case 32:
            return ("personal_information", (1371, 1493), (1386, 1509), 4,
0, 0)
        case 33:
            return ("signature", (150, 1693), (675, 1745), 4, 0, 0)
        case -1:
            return ("empty", (0, 0), (0, 0), -1, -1, -1)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78014_1(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
```

```
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (33, (0, 0), (0, 0), -1, -1)
        case 1:
            return ("family_name", (102, 305), (597, 372), 1, 1, 0)
        case 2:
            return ("given_name", (602, 305), (1097, 372), 1, 1, 0)
        case 3:
            return ("middle_name", (1102, 305), (1594, 372), 1, 1, 0)
        case 4:
            return ("year_of_birth", (127, 430), (271, 498), 0, 2, 0)
        case 5:
            return ("month_of_birth", (302, 430), (421, 498), 0, 2, 0)
        case 6:
            return ("day_of_birth", (452, 430), (571, 498), 0, 2, 0)
        case 7:
            return ("gender_male", (754, 455), (769, 470), 4, 0, 0)
        case 8:
            return ("gender_female", (754, 480), (769, 495), 4, 0, 0)
        case 9:
            return ("gender_X", (754, 505), (769, 520), 4, 0, 0)
        case 10:
            return ("service_number", (828, 430), (1171, 498), 0, 0, 0)
        case 11:
            return ("rank", (1227, 430), (1346, 498), 2, 0, 0)
        case 12:
            return ("language_FR", (1529, 455), (1544, 470), 4, 0, 0)
        case 13:
            return ("language_EN", (1529, 480), (1544, 495), 4, 0, 0)
        case 14:
            return ("number_and_street", (102, 625), (1594, 697), 2, 3, 0)
        case 15:
            return ("apt./unit", (102, 730), (297, 797), 2, 3, 0)
        case 16:
            return ("city", (303, 730), (847, 797), 1, 1, 0)
        case 17:
```

```
            return ("province", (852, 730), (1347, 797), 1, 1, 0)
        case 18:
            return ("postal_code", (1352, 730), (1594, 796), 3, 0, 0)
        case 19:
            return ("lot_and_concession_number", (102, 830), (847, 897), 2,
0, 0)
        case 20:
            return ("section", (852, 830), (1594, 897), 2, 3, 0)
        case 21:
            return ("same_ordinary_address", (938, 945), (961, 968), 4, 0,
0)
        case 22:
            return ("mailing_address", (102, 1005), (1594, 1072), 2, 3, 1)
        case 23:
            return ("mailing_country", (1077, 1105), (1372, 1172), 1, 1, 1)
        case 24:
            return ("mailing_province", (652, 1105), (1072, 1172), 1, 3, 1)
        case 25:
            return ("mailing_city", (102, 1105), (647, 1172), 1, 3, 1)
        case 26:
            return ("mailing_postal_code", (1377, 1105), (1594, 1171), 3, 3,
1)
        case 27:
            return ("mailing_address_unit", (102, 1255), (1594, 1322), 2, 3,
0)
        case 28:
            return ("mailing_country_unit", (1077, 1355), (1372, 1422), 1,
1, 0)
        case 29:
            return ("mailing_province_unit", (652, 1355), (1072, 1422), 1,
3, 0)
        case 30:
            return ("mailing_city_unit", (102, 1355), (647, 1422), 1, 3, 0)
        case 31:
            return ("mailing_postal_code_unit", (1377, 1355), (1594, 1421),
3, 3, 0)
        case 32:
            return ("personal_information", (1504, 1493), (1520, 1509), 4,
0, 0)
        case 33:
            return ("signature", (150, 1693), (675, 1746), 4, 0, 0)
        case -1:
```

```
            return ("empty", (0, 0), (0, 0), -1, -1, -1)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_77010_S(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (15, (0, 0), (0, 0), -1, -1, 0)
        case 1:
            return ("family_name", (127, 351), (872, 397), 1, 1, 0)
        case 2:
            return ("given_name", (127, 451), (872, 497), 1, 1, 0)
        case 3:
            return ("middle_name", (127, 551), (872, 597), 1, 1, 0)
        case 4:
            return ("year_of_birth", (902, 552), (1172, 597), 0, 2, 0)
        case 5:
            return ("month_of_birth", (1177, 552), (1372, 597), 0, 2, 0)
        case 6:
            return ("day_of_birth", (1377, 552), (1572, 597), 0, 2, 0)
        case 7:
            return ("number_and_street", (129, 1000), (1422, 1047), 2, 3, 0)
        case 8:
            return ("apt./unit", (1427, 1000), (1570, 1047), 0, 0, 0)
```

```
        case 9:
            return ("city", (129, 1100), (822, 1147), 1, 1, 0)
        case 10:
            return ("province", (827, 1100), (1297, 1147), 1, 1, 0)
        case 11:
            return ("postal_code", (1302, 1100), (1570, 1147), 3, 3, 0)
        case 12:
            return ("signature", (175, 1350), (624, 1421), 4, 0, 0)
        case 13:
            return ("correctional_institution", (485, 1577), (1570, 1622),
1, 0, 0)
        case 14:
            return ("electoral_district_code", (627, 1627), (872, 1672), 0,
0, 0)
        case 15:
            return ("electoral_district_name", (877, 1627), (1570, 1672), 1,
0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_77010_1S(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (15, (0, 0), (0, 0), -1, -1, 0)
```

```
        case 1:
            return ("family_name", (127, 351), (872, 397), 1, 1, 0)
        case 2:
            return ("given_name", (127, 451), (872, 497), 1, 1, 0)
        case 3:
            return ("middle_name", (127, 551), (872, 597), 1, 1, 0)
        case 4:
            return ("year_of_birth", (902, 552), (1172, 597), 0, 2, 0)
        case 5:
            return ("month_of_birth", (1177, 552), (1372, 597), 0, 2, 0)
        case 6:
            return ("day_of_birth", (1377, 552), (1572, 597), 0, 2, 0)
        case 7:
            return ("number_and_street", (129, 1025), (1422, 1072), 2, 3, 0)
        case 8:
            return ("apt./unit", (1427, 1025), (1570, 1072), 0, 0, 0)
        case 9:
            return ("city", (129, 1125), (822, 1172), 1, 1, 0)
        case 10:
            return ("province", (827, 1125), (1297, 1172), 1, 1, 0)
        case 11:
            return ("postal_code", (1302, 1125), (1570, 1172), 3, 3, 0)
        case 12:
            return ("signature", (175, 1375), (623, 1446), 4, 0, 0)
        case 13:
            return ("correctional_institution", (475, 1577), (1570, 1622),
1, 0, 0)
        case 14:
            return ("electoral_district_code", (702, 1627), (922, 1672), 0,
0, 0)
        case 15:
            return ("electoral_district_name", (927, 1627), (1570, 1672), 1,
0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78500_W(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\
```

```
    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (26, (0, 0), (0, 0), -1, -1, 0)
        case 1:
            return ("family_name", (153, 330), (872, 397), 1, 1, 0)
        case 2:
            return ("given_name", (927, 330), (1547, 397), 1, 1, 0)
        case 3:
            return ("middle_name", (153, 460), (597, 522), 1, 1, 0)
        case 4:
            return ("phone_number", (652, 460), (1022, 522), 0, 0, 0)
        case 5:
            return ("second_phone_number", (1077, 460), (1472, 522), 0, 0,
0)
        case 6:
            return ("gender_male", (254, 606), (267, 618), 4, 0, 0)
        case 7:
            return ("gender_female", (429, 606), (442, 618), 4, 0, 0)
        case 8:
            return ("gender_x", (629, 606), (642, 618), 4, 0, 0)
        case 9:
            return ("year_of_birth", (702, 585), (872, 647), 0, 2, 0)
        case 10:
            return ("month_of_birth", (902, 585), (1022, 647), 0, 2, 0)
        case 11:
            return ("day_of_birth", (1052, 585), (1172, 647), 0, 2, 0)
        case 12:
            return ("language_EN", (1354, 606), (1367, 618), 4, 0, 0)
        case 13:
```

```
            return ("language_FR", (1529, 606), (1542, 618), 4, 0, 0)
        case 14:
            return ("email", (153, 705), (1547, 772), 2, 0, 0)
        case 15:
            return ("number_and_street", (153, 905), (1347, 972), 2, 3, 0)
        case 15:
            return ("apt./unit", (1402, 905), (1547, 972), 0, 0, 0)
        case 16:
            return ("city", (152, 1030), (722, 1097), 1, 1, 0)
        case 17:
            return ("province", (777, 1030), (1222, 1097), 1, 1, 0)
        case 18:
            return ("postal_code", (1279, 1030), (1547, 1097), 3, 3,
0)
        case 19:
            return ("mailing_address", (153, 1235), (1547, 1297), 2, 3, 0)
        case 20:
            return ("mailing_address_2", (153, 1360), (1547, 1422), 2, 3, 0)
        case 21:
            return ("mailing_country", (153, 1610), (698, 1670), 1, 1, 0)
        case 22:
            return ("mailing_province", (777, 1485), (1247, 1545), 1, 3, 0)
        case 23:
            return ("mailing_city", (153, 1485), (722, 1545), 1, 3, 0)
        case 24:
            return ("mailing_postal_code", (752, 1610), (1022, 1670), 2, 3,
0)
        case 25:
            return ("personal_information", (1404, 1781), (1417, 1793), 4,
0, 0)
        case 26:
            return ("signature", (150, 1920), (1199, 1998), 4, 0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78500_1W(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\
```

```
    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (26, (0, 0), (0, 0), -1, -1, 0)
        case 1:
            return ("family_name", (153, 355), (872, 422), 1, 1, 0)
        case 2:
            return ("given_name", (927, 355), (1547, 422), 1, 1, 0)
        case 3:
            return ("middle_name", (153, 480), (597, 547), 1, 1, 0)
        case 4:
            return ("phone_number", (652, 480), (1022, 547), 0, 0, 0)
        case 5:
            return ("second_phone_number", (1077, 480), (1472, 547), 0, 0,
0)
        case 6:
            return ("gender_male", (279, 631), (292, 643), 4, 0, 0)
        case 7:
            return ("gender_female", (454, 631), (467, 643), 4, 0, 0)
        case 8:
            return ("gender_x", (629, 631), (642, 643), 4, 0, 0)
        case 9:
            return ("year_of_birth", (702, 610), (872, 647), 0, 2, 0)
        case 10:
            return ("month_of_birth", (902, 610), (1022, 647), 0, 2, 0)
        case 11:
            return ("day_of_birth", (1052, 610), (1172, 647), 0, 2, 0)
        case 12:
            return ("language_FR", (1354, 631), (1367, 643), 4, 0, 0)
        case 13:
```

```
            return ("language_EN", (1529, 631), (1542, 643), 4, 0, 0)
        case 14:
            return ("email", (153, 710), (1547, 772), 2, 0, 0)
        case 15:
            return ("number_and_street", (153, 910), (1347, 972), 2, 3, 0)
        case 15:
            return ("apt./unit", (1402, 910), (1547, 972), 0, 0, 0)
        case 16:
            return ("city", (153, 1035), (722, 1097), 1, 1, 0)
        case 17:
            return ("province", (777, 1035), (1222, 1097), 1, 1, 0)
        case 18:
            return ("postal_code", (1279, 1035), (1547, 1097), 3, 3,
0)
        case 19:
            return ("mailing_address", (153, 1235), (1547, 1297), 2, 3, 0)
        case 20:
            return ("mailing_address_2", (153, 1360), (1547, 1420), 2, 3, 0)
        case 21:
            return ("mailing_country", (153, 1610), (698, 1670), 1, 1, 0)
        case 22:
            return ("mailing_province", (777, 1485), (1247, 1545), 1, 3, 0)
        case 23:
            return ("mailing_city", (153, 1485), (722, 1545), 1, 3, 0)
        case 24:
            return ("mailing_postal_code", (752, 1610), (1022, 1670), 2, 3,
0)
        case 25:
            return ("personal_information", (1529, 1781), (1542, 1793), 4,
0, 0)
        case 26:
            return ("signature", (150, 1930), (750, 1998), 4, 0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78510(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\
```

```
    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (28, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (152, 360), (872, 422), 1, 1, 0)
        case 2:
            return ("given_name", (927, 360), (1547, 422), 1, 1, 0)
        case 3:
            return ("middle_name", (152, 485), (597, 547), 1, 1, 0)
        case 4:
            return ("phone_number", (652, 485), (1047, 547), 0, 0, 0)
        case 5:
            return ("secondary_phone_number", (1102, 485), (1497, 547), 0,
0, 0)
        case 6:
            return ("gender_male", (254, 629), (270, 645), 4, 0, 0)
        case 7:
            return ("gender_female", (429, 629), (445, 645), 4, 0, 0)
        case 8:
            return ("gender_x", (704, 629), (720, 645), 4, 0, 0)
        case 9:
            return ("year_of_birth", (752, 610), (897, 672), 1, 2, 0)
        case 10:
            return ("month_of_birth", (927, 610), (1047, 672), 1, 2, 0)
        case 11:
            return ("day_of_birth", (1077, 610), (1197, 672), 1, 2, 0)
        case 12:
            return ("language_EN", (1353, 631), (1368, 646), 4, 0, 0)
        case 13:
```

```
            return ("language_FR", (1531, 631), (1546, 646), 4, 0, 0)
        case 14:
            return ("email", (152, 730), (1547, 797), 2, 0, 0)
        case 15:
            return ("number_and_street", (152, 935), (1347, 997), 2, 3, 0)
        case 16:
            return ("apt./unit", (1402, 935), (1547, 997), 0, 0, 0)
        case 17:
            return ("city", (152, 1060), (722, 1122), 1, 1, 0)
        case 18:
            return ("province", (777, 1060), (1222, 1122), 1, 1, 0)
        case 19:
            return ("postal_code", (1277, 1060), (1547, 1122), 3, 3, 0)
        case 20:
            return ("same_home_address", (601, 1194), (624, 1217), 4, 0,
0)
        case 21:
            return ("mailing_number_and_street_1", (152, 1285), (1547,
1347), 2, 3, 1)
        case 22:
            return ("mailing_number_and_street_2", (152, 1410), (947, 1472),
2, 3, 1)
        case 23:
            return ("mailing_country", (677, 1535), (1222, 1597), 1, 0, 1)
        case 24:
            return ("mailing_province", (152, 1535), (622, 1597), 1, 0, 1)
        case 25:
            return ("mailing_city", (977, 1410), (1547, 1472), 1, 1, 1)
        case 26:
            return ("mailing_postal_code", (1277, 1535), (1547, 1597), 3, 3,
1)
        case 27:
            return ("personal_information", (1404, 1704), (1420, 1720), 4,
0, 0)
        case 28:
            return ("signature", (400, 1950), (1012, 1998), 4, 0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78510_1(caseID):
```

```
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (28, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (152, 360), (872, 422), 1, 1, 0)
        case 2:
            return ("given_name", (927, 360), (1547, 422), 1, 1, 0)
        case 3:
            return ("middle_name", (152, 485), (597, 547), 1, 1, 0)
        case 4:
            return ("phone_number", (652, 485), (1047, 547), 0, 0, 0)
        case 5:
            return ("secondary_phone_number", (1102, 485), (1497, 547), 0,
0, 0)
        case 6:
            return ("gender_male", (279, 629), (295, 645), 4, 0, 0)
        case 7:
            return ("gender_female", (454, 629), (470, 645), 4, 0, 0)
        case 8:
            return ("gender_x", (679, 629), (695, 645), 4, 0, 0)
        case 9:
            return ("year_of_birth", (727, 610), (897, 672), 1, 2, 0)
        case 10:
            return ("month_of_birth", (928, 610), (1047, 672), 1, 2, 0)
        case 11:
            return ("day_of_birth", (1077, 610), (1197, 672), 1, 2, 0)
```

```
        case 12:
            return ("language_FR", (1354, 629), (1370, 645), 4, 0, 0)
        case 13:
            return ("language_EN", (1529, 629), (1545, 645), 4, 0, 0)
        case 14:
            return ("email", (152, 730), (1547, 797), 2, 0, 0)
        case 15:
            return ("number_and_street", (152, 935), (1347, 997), 2, 3, 0)
        case 16:
            return ("apt./unit", (1402, 935), (1547, 997), 0, 0, 0)
        case 17:
            return ("city", (152, 1060), (722, 1122), 1, 1, 0)
        case 18:
            return ("province", (777, 1060), (1222, 1122), 1, 1, 0)
        case 19:
            return ("postal_code", (1277, 1060), (1547, 1122), 3, 3, 0)
        case 20:
            return ("same_home_address", (853, 1194), (876, 1217), 4, 0,
0)
        case 21:
            return ("mailing_number_and_street_1", (152, 1285), (1547,
1347), 2, 3, 1)
        case 22:
            return ("mailing_number_and_street_2", (152, 1410), (947, 1472),
2, 3, 1)
        case 23:
            return ("mailing_country", (677, 1535), (1222, 1597), 1, 0, 1)
        case 24:
            return ("mailing_province", (152, 1535), (622, 1597), 1, 0, 1)
        case 25:
            return ("mailing_city", (977, 1410), (1547, 1472), 1, 1, 1)
        case 26:
            return ("mailing_postal_code", (1277, 1535), (1547, 1597), 3, 3,
1)
        case 27:
            return ("personal_information", (1529, 1704), (1545, 1720), 4,
0, 0)
        case 28:
            return ("signature", (400, 1950), (1012, 1998), 4, 4, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
```

```
            exit

def form_78520_W(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (26, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (152, 335), (872, 397), 1, 1, 0)
        case 2:
            return ("given_name", (927, 335), (1547, 397), 1, 1, 0)
        case 3:
            return ("middle_name", (152, 460), (597, 522), 1, 1, 0)
        case 4:
            return ("phone_number", (652, 460), (1022, 522), 0, 0, 0)
        case 5:
            return ("second_phone_number", (1077, 460), (1472, 522), 0, 0,
0)
        case 6:
            return ("gender_male", (254, 606), (267, 618), 4, 0, 0)
        case 7:
            return ("gender_female", (429, 606), (442, 618), 4, 0, 0)
        case 8:
            return ("gender_x", (629, 606), (642, 618), 4, 0, 0)
        case 9:
            return ("year_of_birth", (677, 585), (847, 647), 0, 2, 0)
        case 10:
```

```
            return ("month_of_birth", (877, 585), (997, 647), 0, 2, 0)
        case 11:
            return ("day_of_birth", (1027, 585), (1147, 647), 0, 2, 0)
        case 12:
            return ("language_EN", (1354, 604), (1370, 620), 4, 0, 0)
        case 13:
            return ("language_FR", (1529, 604), (1545, 620), 4, 0, 0)
        case 14:
            return ("email", (152, 705), (1547, 772), 2, 0, 0)
        case 15:
            return ("number_and_street", (152, 910), (1347, 972), 2, 3, 0)
        case 15:
            return ("apt./unit", (1402, 910), (1547, 972), 0, 0, 0)
        case 16:
            return ("city", (153, 1035), (722, 1097), 1, 1, 0)
        case 17:
            return ("province", (777, 1035), (1222, 1097), 1, 1, 0)
        case 18:
            return ("postal_code", (1277, 1035), (1547, 1097), 3, 3,
0)
        case 19:
            return ("same_ordinary_address", (589, 1169), (612, 1192), 4, 0,
0)
        case 20:
            return ("mailing_number_and_street", (152, 1260), (1347, 1322),
3, 3, 1)
        case 21:
            return ("mailing_apt./unit", (1402, 1260), (1547, 1322), 0, 0,
1)
        case 22:
            return ("mailing_province", (752, 1385), (1222, 1447), 1, 1, 1)
        case 23:
            return ("mailing_city", (152, 1385), (697, 1446), 1, 1, 1)
        case 24:
            return ("mailing_postal_code", (1277, 1385), (1547, 1447), 3, 3,
1)
        case 25:
            return ("personal_information", (1404, 1546), (1420, 1561), 4,
0, 0)
        case 26:
            return ("signature", (375, 1780), (1025, 1823), 4, 0, 0)
        case _:
```

```
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78520_1W(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (26, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (152, 335), (872, 397), 1, 1, 0)
        case 2:
            return ("given_name", (927, 335), (1547, 397), 1, 1, 0)
        case 3:
            return ("middle_name", (152, 460), (597, 522), 1, 1, 0)
        case 4:
            return ("phone_number", (652, 460), (1047, 522), 0, 0, 0)
        case 5:
            return ("second_phone_number", (1102, 460), (1497, 522), 0, 0,
0)
        case 6:
            return ("gender_male", (279, 606), (292, 618), 4, 0, 0)
        case 7:
            return ("gender_female", (454, 606), (467, 618), 4, 0, 0)
        case 8:
            return ("gender_x", (629, 606), (642, 618), 4, 0, 0)
        case 9:
```

```
            return ("year_of_birth", (702, 585), (872, 647), 0, 2, 0)
        case 10:
            return ("month_of_birth", (902, 585), (1022, 647), 0, 2, 0)
        case 11:
            return ("day_of_birth", (1052, 585), (1172, 647), 0, 2, 0)
        case 12:
            return ("language_FR", (1354, 604), (1370, 620), 4, 0, 0)
        case 13:
            return ("language_EN", (1529, 604), (1545, 620), 4, 0, 0)
        case 14:
            return ("email", (152, 705), (1547, 772), 2, 0, 0)
        case 15:
            return ("number_and_street", (152, 910), (1347, 972), 2, 3, 0)
        case 15:
            return ("apt./unit", (1402, 910), (1547, 972), 0, 0, 0)
        case 16:
            return ("city", (153, 1035), (722, 1097), 1, 1, 0)
        case 17:
            return ("province", (777, 1035), (1222, 1097), 1, 1, 0)
        case 18:
            return ("postal_code", (1277, 1035), (1547, 1097), 3, 3,
0)
        case 19:
            return ("same_ordinary_address", (857, 1169), (880, 1192), 4, 0,
0)
        case 20:
            return ("mailing_number_and_street", (152, 1260), (1347, 1322),
3, 3, 1)
        case 21:
            return ("mailing_apt./unit", (1402, 1260), (1547, 1322), 0, 0,
1)
        case 22:
            return ("mailing_province", (752, 1385), (1222, 1447), 1, 1, 1)
        case 23:
            return ("mailing_city", (152, 1385), (697, 1446), 1, 1, 1)
        case 24:
            return ("mailing_postal_code", (1277, 1385), (1547, 1447), 3, 3,
1)
        case 25:
            return ("personal_information", (1529, 1546), (1545, 1561), 4,
0, 0)
        case 26:
```

```
            return ("signature", (375, 1790), (1025, 1828), 4, 0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78530(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\

    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (24, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (123, 305), (683, 356), 1, 1, 0)
        case 2:
            return ("given_name", (724, 305), (1127, 356), 1, 1, 0)
        case 3:
            return ("middle_name", (1164, 305), (1565, 356), 1, 1, 0)
        case 4:
            return ("gender_male", (209, 444), (229, 464), 4, 0, 0)
        case 5:
            return ("gender_female", (352, 444), (373, 464), 4, 0, 0)
        case 6:
            return ("gender_x", (566, 444), (587, 464), 4, 0, 0)
        case 7:
            return ("year_of_birth", (624, 425), (773, 475), 0, 2, 0)
        case 8:
            return ("month_of_birth", (794, 425), (909, 475), 0, 2, 0)
```

```
        case 9:
            return ("day_of_birth", (930, 425), (1045, 475), 0, 2, 0)
        case 10:
            return ("phone_number", (1089, 425), (1565, 475), 0, 0, 0)
        case 11:
            return ("email", (122, 545), (1565, 597), 2, 0, 0)
        case 12:
            return ("number_and_street", (123, 740), (819, 811), 2, 3, 0)
        case 13:
            return ("apt./unit", (123, 855), (249, 908), 0, 0, 0)
        case 14:
            return ("city", (270, 855), (819, 908), 1, 1, 0)
        case 15:
            return ("province", (123, 970), (501, 1014), 1, 1, 0)
        case 16:
            return ("postal_code", (519, 970), (817, 1015), 3, 3, 0)
        case 17:
            return ("same_home_address", (1273, 650), (1297, 674), 4, 0, 0)
        case 18:
            return ("mailing_address", (874, 765), (1565, 811), 2, 3, 1)
        case 19:
            return ("mailing_apt./unit", (874, 855), (1001, 908), 0, 1, 1)
        case 20:
            return ("mailing_country", (874, 1070), (1249, 1119), 1, 1, 1)
        case 21:
            return ("mailing_province", (874, 970), (1565, 1016), 1, 1, 1)
        case 22:
            return ("mailing_city", (1020, 855), (1565, 908), 1, 1, 1)
        case 23:
            return ("mailing_postal_code", (1266, 1070), (1565, 1119), 3, 0,
1)
        case 24:
            return ("signature", (122, 1305), (924, 1359), 4, 0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_78530_1(caseID):
    '''
    /!\ FONCTION UTILISÉ PAR get_form_case, NE PAS UTILISER MANUELLEMENT /!\
```

```
    La case 0 contient le nombre des cases présent dans le formulaire, les
autres valeurs sont à zéro par défaut
    servent à déterminer la position de la case
    Entrée: Numéro de la case désiré
    sortie: Tuple contenant le contenue de la case, sa position pour
l'extraction ((xi, yi), (xf, yf)), le type (0: Chiffre, 1: Lettre,
2:Chiffre+Lettre,
    3:Alternance (L, C), 4:Case/Signature) et le type de correction (0:
Aucune, 1: Dictionnaire, 2:Date, 3: Addresse API), Skip si même adresse (0:
Non, 1:Oui)
    Par: Marc-Antoine Coulombe
    Par: Marc-Antoine Coulombe
    Date: 2023-11-14
    '''
    match caseID:
        case 0:
            return (24, (0, 0), (0, 0), -1, -1, -1)
        case 1:
            return ("family_name", (123, 305), (683, 356), 1, 1, 0)
        case 2:
            return ("given_name", (724, 305), (1127, 356), 1, 1, 0)
        case 3:
            return ("middle_name", (1164, 305), (1565, 356), 1, 1, 0)
        case 4:
            return ("gender_male", (209, 444), (229, 464), 4, 0, 0)
        case 5:
            return ("gender_female", (393, 444), (414, 464), 4, 0, 0)
        case 6:
            return ("gender_x", (568, 444), (589, 464), 4, 0, 0)
        case 7:
            return ("year_of_birth", (624, 425), (773, 475), 0, 2, 0)
        case 8:
            return ("month_of_birth", (794, 425), (909, 475), 0, 2, 0)
        case 9:
            return ("day_of_birth", (930, 425), (1045, 475), 0, 2, 0)
        case 10:
            return ("phone_number", (1089, 425), (1565, 475), 0, 0, 0)
        case 11:
            return ("email", (122, 545), (1565, 597), 2, 0, 0)
        case 12:
            return ("number_and_street", (123, 740), (819, 789), 2, 3, 0)
        case 13:
```

```
            return ("apt./unit", (123, 855), (249, 908), 0, 0, 0)
        case 14:
            return ("city", (270, 855), (819, 908), 1, 1, 0)
        case 15:
            return ("province", (123, 970), (501, 1016), 1, 1, 0)
        case 16:
            return ("postal_code", (519, 970), (817, 1017), 3, 3, 0)
        case 17:
            return ("same_home_address", (1524, 650), (1548, 674), 4, 0, 0)
        case 18:
            return ("mailing_address", (873, 740), (1565, 789), 2, 3, 1)
        case 19:
            return ("mailing_apt./unit", (874, 855), (1001, 908), 0, 1, 1)
        case 20:
            return ("mailing_country", (874, 1070), (1249, 1119), 1, 1, 1)
        case 21:
            return ("mailing_province", (874, 970), (1565, 1019), 1, 1, 1)
        case 22:
            return ("mailing_city", (1020, 855), (1565, 908), 1, 1, 1)
        case 23:
            return ("mailing_postal_code", (1266, 1070), (1565, 1119), 3, 0,
1)
        case 24:
            return ("signature", (122, 1305), (924, 1359), 4, 0, 0)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de case non-reconnu" +
bcolors.ENDC)
            exit

def form_439_F_1(caseID):
    '''/!\\ for testing only, do not use /!\\'''
    match caseID:
        case 0:
            return ("2", (0, 0), (0, 0))
        case 1:
            return ("Prenom", (220, 590), (879, 667))
        case 2:
            return ("Nom", (880, 590), (1600, 667))
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de formulaire non-reconnu"
+ bcolors.ENDC)
            exit
```

```
def get_form_case(formID, caseID):
    casePosition = ()
    match formID:
        case "77010-S":
            casePosition = form_77010_S(caseID)
        case "77010-1S":
            casePosition = form_77010_1S(caseID)
        case "78014":
            casePosition = form_78014(caseID)
        case "78014-1":
            casePosition = form_78014_1(caseID)
        case "78500-W":
            casePosition = form_78500_W(caseID)
        case "78500-1W":
            casePosition = form_78500_1W(caseID)
        case "78510":
            casePosition = form_78510(caseID)
        case "78510-1":
            casePosition = form_78510_1(caseID)
        case "78520-W":
            casePosition = form_78520_W(caseID)
        case "78520-1W":
            casePosition = form_78520_1W(caseID)
        case "78530":
            casePosition = form_78530(caseID)
        case "78530-1":
            casePosition = form_78530_1(caseID)
        case "439-F-1":
            casePosition = form_439_F_1(caseID)
        case _:
            print(bcolors.FAIL + "Erreur: Numéro de formulaire non-reconnu"
+ bcolors.ENDC)
            exit
    return casePosition
```

ANNEXE VIII : PROGRAMME GÉNÉRANT LES TRANSFORMÉES

```
'''
Nom du programme : Programme générant les transformées
Nom du fichier : transform.py
Contenu du fichier : Fonctions pour les transformées DCT, DWT et Hough
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import utils
import cv2
import numpy as np
import math
from matplotlib import pyplot as plt
import pywt
import pretraitement

def DCT_line(img, save_path=None):
    '''
    /!\ FONCTION UTILISÉ PAR generate_dct, NE PAS UTILISER MANUELLEMENT /!\
    '''
    plt.imshow(img, cmap="gray")
    plt.axis('off')

    dpiImage = 15.2
    if save_path is not None:
        plt.savefig(save_path, bbox_inches='tight', pad_inches = 0, dpi =
dpiImage)
    plt.close('all')
    #plt.show()
    return None

def DWT_line(img, save_path=None):
    '''
    /!\ FONCTION UTILISÉ PAR generate_dwt_dataset, NE PAS UTILISER
MANUELLEMENT /!\
RÉFÉRENCE : Gregory R. Lee, Ralf Gommers, Filip Wasilewski, Kai Wohlfahrt,
Aaron O'Leary (2019). PyWavelets: A Python package for wavelet analysis.
Journal of Open Source Software, 4(36), 1237,
https://doi.org/10.21105/joss.01237.
    '''
    coeffs = pywt.dwt2(img, 'haar')
```

```
    LL, (LH, HL, HH) = coeffs
    #cv2.imshow('Image coeffs', coeffs)
    fig = plt.figure(figsize=(1, 1))
    for i, a in enumerate([LL, LH, HL, HH]):
        ax = fig.add_subplot(2, 2, i+1)
        ax.imshow(a, interpolation="nearest", cmap='gray')
        ax.set_xticks([])
        ax.set_yticks([])
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.spines['bottom'].set_visible(False)
        ax.spines['left'].set_visible(False)
    plt.subplots_adjust(wspace=0, hspace=0)
    if save_path is not None:
        plt.savefig(save_path, bbox_inches='tight', pad_inches = 0, dpi=73)
    plt.close('all')
    return coeffs

def generate_dct(img, debug = False, savePath=None):
    '''
    Fonction permettant de générer la DCT à partir de la base de données
EMNIST
    Entrée: Chemin d'accès vers le répertoire contenant les images EMNIST,
le chemin d'accès pour la sauvegarde
    Sortie: Sauvegarde des images sur le disque
    Par: Marc-Antoine
    Dernière modification: 2023-10-04
    '''
    temporarySavePath = "D:\\Maitrise\\debug\\tempFile_DCT.tiff"
    img = utils.normalizefloat32(img)
    imgDCT = cv2.dct(img, cv2.DCT_INVERSE)
    DCT_line(imgDCT, save_path=temporarySavePath)
    imgDCT = cv2.imread(temporarySavePath, cv2.IMREAD_GRAYSCALE)

    if debug:
        imgResized = pretraitement.img_resize_scale(img, 3)
        cv2.imshow("img", imgResized)
        imgDCTResized = pretraitement.img_resize_scale(imgDCT, 3)
        cv2.imshow("imgDCT", imgDCTResized)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
    if savePath is not None:
```

```
        cv2.imwrite(savePath, imgDCT)
    return imgDCT

def generate_dwt(img, debug = False, savePath=None):
    '''
    Fonction permettant de générer la DWT à partir de la base de données
EMNIST
    Entrée: Chemin d'accès vers le répertoire contenant les images EMNIST,
le chemin d'accès pour la sauvegarde
    Sortie: Sauvegarde des images sur le disque
    Par: Marc-Antoine
    Dernière modification: 2023-10-04
    '''
    temporarySavePath = "D:\\Maitrise\\debug\\tempFile_DWT.tiff"
    imgu32 = utils.normalizeuint32(img)
    DWT_line(imgu32, save_path=temporarySavePath)
    imgDWT = cv2.imread(temporarySavePath, cv2.IMREAD_GRAYSCALE)

    if debug:
        imgResized = pretraitement.img_resize_scale(img, 3)
        cv2.imshow("img", imgResized)
        imgDWTResized = pretraitement.img_resize_scale(imgDWT, 3)
        cv2.imshow("imgDCT", imgDWTResized)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
    if savePath is not None:
        cv2.imwrite(savePath, imgDWT)
    return imgDWT

def hough_line(img, angle_step=1, lines_are_white=True, value_threshold=50):
    """
    /!\ FONCTION UTILISÉ PAR generate_hough_dataset, NE PAS UTILISER
MANUELLEMENT /!\
    Fonction permettant de calculer la transformée de Hough
    Entrée:
    img - 2D binary image with nonzeros representing edges
    angle_step - Spacing between angles to use every n-th angle
                 between -90 and 90 degrees. Default step is 1.
    lines_are_white - boolean indicating whether lines to be detected are
white
    value_threshold - Pixel values above or below the value_threshold are
edges
```

```
    Sortie:
    accumulator - 2D array of the hough transform accumulator
    theta - array of angles used in computation, in radians.
    rhos - array of rho values. Max size is 2 times the diagonal
           distance of the input image.
    Dernière modification: 2023-10-04
    RÉFÉRENCE:
https://github.com/alyssaq/hough_transform/blob/master/hough_transform.py
    SOUS-LICENSE MIT: https://alyssaq.github.io/mit-license/
    """
    # Rho and Theta ranges
    thetas = np.deg2rad(np.arange(-90.0, 90.0, angle_step))
    width, height = img.shape
    diag_len = int(round(math.sqrt(width * width + height * height)))
    rhos = np.linspace(-diag_len, diag_len, diag_len * 2)

    # Cache some resuable values
    cos_t = np.cos(thetas)
    sin_t = np.sin(thetas)
    num_thetas = len(thetas)

    # Hough accumulator array of theta vs rho
    accumulator = np.zeros((2 * diag_len, num_thetas), dtype=np.uint32)
    # (row, col) indexes to edges
    are_edges = img > value_threshold if lines_are_white else img <
value_threshold
    y_idxs, x_idxs = np.nonzero(are_edges)

    # Vote in the hough accumulator
    for i in range(len(x_idxs)):
        x = x_idxs[i]
        y = y_idxs[i]

        for t_idx in range(num_thetas):
            # Calculate rho. diag_len is added for a positive index
            rho = diag_len + int(round(x * cos_t[t_idx] + y * sin_t[t_idx]))
            accumulator[rho, t_idx] += 1

    return accumulator, thetas, rhos

def show_hough_line(accumulator, thetas, rhos, save_path=None):
    '''
```

```
    /!\ FONCTION UTILISÉ PAR generate_hough_dataset, NE PAS UTILISER
MANUELLEMENT /!\
    '''
    plt.imshow(accumulator, aspect='auto', cmap='gray',
extent=[np.rad2deg(thetas[-1]), np.rad2deg(thetas[0]), rhos[-1], rhos[0]])
    plt.axis('off')
    if save_path is not None:
        plt.savefig(save_path, bbox_inches='tight', pad_inches = 0,
dpi=11.3)
    plt.close('all')
    return None

def generate_hough(img, debug = False, savePath = None):
    '''
    Fonction permettant de générer les accumulateurs de Hough d'une image
    Entrée: image, bool debug, le chemin d'accès pour la sauvegarde
    Sortie: Sauvegarde des images sur le disque
    Par: Marc-Antoine
    Dernière modification: 2023-10-04
    '''
    temporarySavePath = "D:\\Maitrise\\debug\\tempFile_Hough.tiff"
    img = np.array([img])[0,:,:]
    accumulator, thetas, rhos = hough_line(img)
    show_hough_line(accumulator, thetas, rhos, save_path=temporarySavePath)
    imgHough = cv2.imread(temporarySavePath, cv2.IMREAD_GRAYSCALE)

    if debug:
        imgResized = pretraitement.img_resize_scale(img, 3)
        cv2.imshow("img", imgResized)
        imgHoughResized = pretraitement.img_resize_scale(imgHough, 3)
        cv2.imshow("imgDCT", imgHoughResized)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
    if savePath is not None:
        cv2.imwrite(savePath, imgHough)
    return imgHough

def generate_all_transforms(imgCharacter, debug = False):
    '''
    Fonction permettant de générer l'ensemble des transformées en un appel
    Entrée: image, bool debug
    Sortie: Sauvegarde des images sur le disque
```

```
Par: Marc-Antoine
Dernière modification: 2023-10-04
'''
imgCharacter = utils.normalizeuint8(imgCharacter)
imgCharacterDCT = generate_dct(imgCharacter, debug = debug)
imgCharacterDWT = generate_dwt(imgCharacter, debug = debug)
imgCharacterHough = generate_hough(imgCharacter, debug = debug)
return imgCharacter, imgCharacterDCT, imgCharacterDWT, imgCharacterHough
```

ANNEXE IX : PROGRAMME DE CLASSIFICATION

```
'''
Nom du programme : Programme de classification
Nom du fichier : tflite.py
Contenu du fichier : Fonctions pour le chargement et évaluation des réseaux
de neurones
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import tensorflow as tf
from utils import bcolors
import numpy as np
import utils

def generate_TfLite_model(savedModelPath, tfliteModelPath):
    '''
    Fonction qui génère le modèle Tensorflow Lite à partir d'un modèle
entraîné
    Entrée: Chemin d'accès du modèle entraîné, chemin d'accès pour la
sauvegarde du modèle Tensorflow Lite
    Sortie: Sauvegarde le modèle Tensorflow Lite sur le disque
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    saved_model_path = savedModelPath
    model = tf.keras.models.load_model(saved_model_path)

    converter = tf.lite.TFLiteConverter.from_saved_model(saved_model_path)

    tflite_model = converter.convert()

    tflite_model_path = tfliteModelPath
    tf.io.write_file(tflite_model_path, tflite_model)
    return None

def load_TfLite_model(tflite_model_path):
    '''
    Entrée: Chemin d'accès vers le modèle Tensorflow Lite
    Sortie: interpreter, input_details, output_details
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
```

```
    '''
    # Load the TFLite model and allocate tensors.
    interpreter = tf.lite.Interpreter(model_path=tflite_model_path)
    interpreter.allocate_tensors()

    # Get input and output tensors.
    input_details = interpreter.get_input_details()
    output_details = interpreter.get_output_details()

    return [interpreter, input_details, output_details]

def invoke_TfLite_model(inputData, interpreter, input_details,
output_details):
    '''
    Fonction permettant d'évaluer un réseau de neurones
    Entrée: Image de caractères, sortie de la fonction load_TfLite_model
    Sortie: Prédiction du modèle
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    input_shape = input_details[0]['shape']
    interpreter.set_tensor(input_details[0]['index'], inputData)
    interpreter.invoke()

    # The function `get_tensor()` returns a copy of the tensor data.
    # Use `tensor()` in order to get a pointer to the tensor.
    output_data = interpreter.get_tensor(output_details[0]['index'])
    return output_data

def load_all_numbers_models(modelPath):
    '''
    Fonction permettant de retourner quatre listes contenants les modèles de
chiffres
    Entrée: Chemin d'accès vers le répertoire contenant les modèles
    Sortie: Quatre listes contenants les différents modèles
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    classeDebut = 0
    nombreClasse = 10
    modelsOriginal = []
    modelsDCT = []
```

```
    modelsDWT = []
    modelsHough = []
    for i in range(classeDebut, nombreClasse):
        modelsOriginal.append(load_TfLite_model(f'{modelPath}\\reseau_Chiffr
es_{i}.model'))
        modelsDCT.append(load_TfLite_model(f'{modelPath}\\reseau_Chiffres_{i
}_DCT.model'))
        modelsDWT.append(load_TfLite_model(f'{modelPath}\\reseau_Chiffres_{i
}_DWT.model'))
        modelsHough.append(load_TfLite_model(f'{modelPath}\\reseau_Chiffres_
{i}_Hough.model'))

    return modelsOriginal, modelsDCT, modelsDWT, modelsHough

def evaluate_all_numbers_models(img, models, debug = False):
    '''
    Fonction permettant de retourner les évaluations de chacun des modèles
de chiffres
    Entrée: Image à classifier, liste contenant les listes de modèles, mode
deboggage
    Sortie: Liste contenant les prédictions de chacun des réseaux
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    predictions = []
    img = utils.normalizefloat32(img)
    img = np.array([img])
    for modelNumber in range (len(models)):
        try:
            predictions = np.append(predictions,
                invoke_TfLite_model(img, models[modelNumber][0],
models[modelNumber][1], models[modelNumber][2]))
        except:
            print(bcolors.FAIL + "Erreur détectée, ne peut itérer: Arrêt du
programme" + bcolors.ENDC)
            exit()
    if debug:
        print("Predictions are: ", predictions)
    return predictions

def load_all_letters_models(modelPath):
    '''
```

```
    Fonction permettant de retourner quatre listes contenants les modèles de
lettres
    Entrée: Chemin d'accès vers le répertoire contenant les modèles
    Sortie: Quatre listes contenants les différents modèles
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    classeDebut = 0 # Ne commence pas à à zéro, mais la liste est indexé à
zéro
    nombreClasse = 26 # Ne commence pas à à zéro, mais la liste est indexé à
zéro
    classesListMaj = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 53, 54]
    classesListMin = [27, 28, 3, 30, 31, 32, 33, 34, 35, 36, 11, 38, 13, 40,
15, 16, 43, 44, 19, 46, 21, 22, 23, 24, 25, 26, 53, 54]

    modelsOriginalMaj = []
    modelsDCTMaj = []
    modelsDWTMaj = []
    modelsHoughMaj = []
    for classe in classesListMaj:
        modelsOriginalMaj.append(load_TfLite_model(f'{modelPath}\\reseau_Cla
sses_{classe}.model'))
        modelsDCTMaj.append(load_TfLite_model(f'{modelPath}\\reseau_Classes_
{classe}_DCT.model'))
        modelsDWTMaj.append(load_TfLite_model(f'{modelPath}\\reseau_Classes_
{classe}_DWT.model'))
        modelsHoughMaj.append(load_TfLite_model(f'{modelPath}\\reseau_Classe
s_{classe}_Hough.model'))

    modelsOriginalMin = []
    modelsDCTMin = []
    modelsDWTMin = []
    modelsHoughMin = []
    for classe in classesListMin:
        modelsOriginalMin.append(load_TfLite_model(f'{modelPath}/reseau_Clas
ses_{classe}.model'))
        modelsDCTMin.append(load_TfLite_model(f'{modelPath}/reseau_Classes_{
classe}_DCT.model'))
        modelsDWTMin.append(load_TfLite_model(f'{modelPath}/reseau_Classes_{
classe}_DWT.model'))
```

```
        modelsHoughMin.append(load_TfLite_model(f'{modelPath}/reseau_Classes
_{classe}_Hough.model'))

    modelsOriginal = [modelsOriginalMaj, modelsOriginalMin]
    modelsDCT = [modelsDCTMaj, modelsDCTMin]
    modelsDWT = [modelsDWTMaj, modelsDWTMin]
    modelsHough = [modelsHoughMaj, modelsHoughMin]

    return modelsOriginal, modelsDCT, modelsDWT, modelsHough

def evaluate_all_letters_models(img, models, debug = False):
    '''
    Fonction permettant de retourner les évaluations de chacun des modèles
de lettres
    Entrée: Image à classifier, liste contenant les listes de modèles, mode
deboggage
    Sortie: Liste contenant les prédictions de chacun des réseaux
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    predictions = []
    modelsMaj = models[0]
    modelsMin = models[1]

    modelDistincts = [27, 28, 30, 31, 32, 33, 34, 35, 36, 38, 40, 43, 44,
46]

    img = utils.normalizefloat32(img)
    img = np.array([img])

    for modelNumber in range (len(modelsMaj)):
        try:
            if (modelNumber + 27) not in modelDistincts:
                predictions = np.append(predictions,
                    invoke_TfLite_model(img, modelsMaj[modelNumber][0],
modelsMaj[modelNumber][1], modelsMaj[modelNumber][2]))
            else:
                sommePredictionMaj = invoke_TfLite_model(img,
modelsMaj[modelNumber][0],
                                                         modelsMaj[modelNumber][1],
modelsMaj[modelNumber][2])
```

```
                sommePredictionMin = invoke_TfLite_model(img,
modelsMin[modelNumber][0],
                                                         modelsMin[modelNumber][1],
modelsMin[modelNumber][2])
                if sommePredictionMaj >= sommePredictionMin:
                    sommePrediction = sommePredictionMaj
                else:
                    sommePrediction = sommePredictionMin
                #sommePrediction = (sommePredictionMaj+sommePredictionMin)/2
                predictions = np.append(predictions, sommePrediction)
        except:
            print(bcolors.FAIL + "Erreur détectée, ne peut itérer: Arrêt du
programme" + bcolors.ENDC)
            exit()

    if debug:
        print("Predictions are: ", predictions)

    return predictions

def load_all_mixte_models(modelPath):
    '''
    Fonction permettant de retourner quatre listes contenants les modèles
mixtes (lettres+chiffres)
    Entrée: Chemin d'accès vers le répertoire contenant les modèles
    Sortie: Quatre listes contenants les différents modèles
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    modelsOriginal = []
    modelsDCT = []
    modelsDWT = []
    modelsHough = []
    modelsOriginal.append(load_TfLite_model(f'{modelPath}\\reseau_Mixte.mode
l'))
    modelsDCT.append(load_TfLite_model(f'{modelPath}\\reseau_Mixte_DCT.model
'))
    modelsDWT.append(load_TfLite_model(f'{modelPath}\\reseau_Mixte_DWT.model
'))
    modelsHough.append(load_TfLite_model(f'{modelPath}\\reseau_Mixte_Hough.m
odel'))
```

```
    return modelsOriginal, modelsDCT, modelsDWT, modelsHough

def evaluate_all_mixte_models(img, models, debug = False):
    '''
    Fonction permettant de retourner les évaluations de chacun des modèles
mixtes (lettres+chiffres)
    Entrée: Image à classifier, liste contenant les listes de modèles, mode
deboggage
    Sortie: Liste contenant les prédictions de chacun des réseaux
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    predictions = []
    img = utils.normalizefloat32(img)
    img = np.array([img])
    for modelNumber in range (len(models)):
        try:
            predictions = np.append(predictions,
                invoke_TfLite_model(img, models[modelNumber][0],
models[modelNumber][1], models[modelNumber][2]))
        except:
            print(bcolors.FAIL + "Erreur détectée, ne peut itérer: Arrêt du
programme" + bcolors.ENDC)
            exit()
    if debug:
        print("Predictions are: ", predictions)
    return predictions

def return_character(predictions, characType, seuil = 0.01, debug = False):
    '''
    Fonction permettant de retourner le chiffre ou la lettre selon
l'évaluation
    Entrée: Liste des prédictions, type de charactère, seuil minimum de
détection,
        mode deboggage
    Sortie: charactère reconnu
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    if(characType == 0): #is a number
        prediction1 = predictions.argsort()[-1]
        prediction2 = predictions.argsort()[-2]
```

```
        predictionValue = predictions[prediction1]
        character = str(prediction1)
        if(predictionValue < seuil):
            character = "?"
    elif(characType == 1): # is a letter
        prediction1 = predictions.argsort()[-1] + 1
        prediction2 = predictions.argsort()[-2] + 1
        predictionValue = predictions[prediction1 - 1]
        if(predictionValue < seuil):
            prediction1 = "?"
        lettersIndex = {
            "?" : "?",
            1 : "A",
            2 : "B",
            3 : "C",
            4 : "D",
            5 : "E",
            6 : "F",
            7 : "G",
            8 : "H",
            9 : "I",
            10 : "J",
            11 : "K",
            12 : "L",
            13 : "M",
            14 : "N",
            15 : "O",
            16 : "P",
            17 : "Q",
            18 : "R",
            19 : "S",
            20 : "T",
            21 : "U",
            22 : "V",
            23 : "W",
            24 : "X",
            25 : "Y",
            26 : "Z",
            27 : "-",
            28 : "@"
        }
        character = lettersIndex[prediction1]
```

```
    elif(characType == 2): #is a mixte of numbers/letters
        prediction1 = predictions.argsort()[-1]
        predictionValue = predictions[prediction1]
        character = prediction1
        if(predictionValue < seuil):
            character = "?"

    else:
        print(bcolors.FAIL + "ERREUR DANS LA DÉTERMINATION DU CARACTÈRE
SELON L'INDEX." + bcolors.ENDC)
        exit()

    if debug:
        print("Letter detected is: ", character)

    return character, predictionValue

def prediction_numbers(imgCharacter, modelsOriginalChiffres,
imgCharacterDCT, modelsDCTChiffres, imgCharacterDWT, modelsDWTChiffres,
imgCharacterHough, modelsHoughChiffres, debug = False):
    '''
    Fonction retournant le chiffre final prédit selon chacune des
caractéristiques
    Entrée: Image chiffre, modèles chiffres originaux, image DCT,
        modèles chiffres DCT, image DWT, modèles chiffres DWT, image Hough,
        modèles chiffres Hough, deboggage
    Sortie: Chiffre reconnu, poids
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    predChiffresImgOriginal = evaluate_all_numbers_models(imgCharacter,
modelsOriginalChiffres, debug = False)
    predChiffresImgDCT  = evaluate_all_numbers_models(imgCharacterDCT,
modelsDCTChiffres, debug = False)
    predChiffresImgDWT  = evaluate_all_numbers_models(imgCharacterDWT,
modelsDWTChiffres, debug = False)
    predChiffresImgHough  = evaluate_all_numbers_models(imgCharacterHough,
modelsHoughChiffres, debug = False)
    predChiffres = ponderation(predChiffresImgOriginal, predChiffresImgDCT,
predChiffresImgDWT, predChiffresImgHough, characType = 0)
    charac, characPred = return_character(predChiffres, characType=0)
```

```
    if debug:
        print("Les poids sont: ", predChiffres)
        print("Le caractère détecté est: ", charac)
        print("Le poids est: ", characPred)
    return charac, characPred

def prediction_letters(imgCharacter, modelsOriginalLettres, imgCharacterDCT,
modelsDCTLettres, imgCharacterDWT, modelsDWTLettres, imgCharacterHough,
modelsHoughLettres, debug = False):
    '''
    Fonction retournant la lettre finale prédite selon chacune des
caractéristiques
    Entrée: Image lettre, modèles lettres originales, image DCT,
        modèles lettres DCT, image DWT, modèles lettres DWT, image Hough,
        modèles lettres Hough, deboggage
    Sortie: Lettre reconnue, poids
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    predLettresImgOriginal = evaluate_all_letters_models(imgCharacter,
modelsOriginalLettres, debug = False)
    predLettresImgDCT  = evaluate_all_letters_models(imgCharacterDCT,
modelsDCTLettres, debug = False)
    predLettresImgDWT  = evaluate_all_letters_models(imgCharacterDWT,
modelsDWTLettres, debug = False)
    predLettresImgHough  = evaluate_all_letters_models(imgCharacterHough,
modelsHoughLettres, debug = False)
    predLettres = ponderation(predLettresImgOriginal, predLettresImgDCT,
predLettresImgDWT, predLettresImgHough, characType = 1)
    charac, characPred = return_character(predLettres, characType=1)
    if debug:
        print("Les poids sont original: ", predLettresImgOriginal)
        print("Les poids sont DCT: ", predLettresImgDCT)
        print("Les poids sont DWT: ", predLettresImgDWT)
        print("Les poids sont Hough: ", predLettresImgHough)
        print("Les poids sont: ", predLettres)
        print("Le caractère détecté est: ", charac)
        print("Le poids est: ", characPred)
    return charac, characPred
```

```
def prediction_mixte(imgCharacter, modelsOriginalMixte, imgCharacterDCT,
modelsDCTMixte, imgCharacterDWT, modelsDWTMixte, imgCharacterHough,
modelsHoughMixte, debug = False):
    '''
    Fonction retournant le type prédit selon chacune des caractéristiques
    Entrée: Image caractère, modèles caractère originales, image DCT,
        modèles caractère DCT, image DWT, modèles caractère DWT, image
Hough,
        modèles caractère Hough, deboggage
    Sortie: Type reconnu, poids
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    predMixteImgOriginal = evaluate_all_mixte_models(imgCharacter,
modelsOriginalMixte, debug = debug)
    predMixteImgDCT  = evaluate_all_mixte_models(imgCharacterDCT,
modelsDCTMixte, debug = debug)
    predMixteImgDWT  = evaluate_all_mixte_models(imgCharacterDWT,
modelsDWTMixte, debug = debug)
    predMixteImgHough  = evaluate_all_mixte_models(imgCharacterHough,
modelsHoughMixte, debug = debug)
    predMixte = ponderation(predMixteImgOriginal, predMixteImgDCT,
predMixteImgDWT, predMixteImgHough, characType = 2)
    predType, predTypeValue = return_character(predMixte, characType=2)
    if debug:

        print("Le type détecté est: ", predType)
        print("Les poids sont: ", predMixte)
    return predType, predTypeValue
```

ANNEXE X : PROGRAMME DE FENÊTRAGE

```
'''
Nom du programme : Programme de fenêtrage
Nom du fichier : window.py
Contenu du fichier : Fonctions pour la segmentation de caractères à l'aide
du glissement d'une fenêtre
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import cv2
from utils import bcolors
import tflite
import pretraitement
import time
import transform

def pyramid(image, scale=1.5, minSize=(30, 30)):
    '''
    Fonction permettant de réduire la taille de l'image afin de retenter de
glisser une fenêtre
    Entrée: image, facteur d'échelle et taille minimale de l'image
    Sortie: NONE
    Par: Marc-Antoine Coulombe
    Dernière modification: 2024-03-31
    Référence: https://pyimagesearch.com/2015/03/23/sliding-windows-for-
object-detection-with-python-and-opencv/
    '''
    # yield the original image
    yield image
    # keep looping over the pyramid
    while True:
        # compute the new dimensions of the image and resize it
        w = int(image.shape[1] / scale)
        image = pretraitement.img_resize_width(image, width=w)
        # if the resized image does not meet the supplied minimum
        # size, then stop constructing the pyramid
        if image.shape[0] < minSize[1] or image.shape[1] < minSize[0]:
            break
        # yield the next image in the pyramid
        yield image
```

```
def sliding_window(image, stepSize, windowSize):
    '''
    Fonction permettant de faire glisser une fenêtre le long d'une image
    Entrée: Image, taille du pas, taille de la fenêtre
    Sortie: NONE
    Par: Marc-Antoine Coulombe
    Dernière modification: 2024-03-31
    Référence: https://pyimagesearch.com/2015/03/23/sliding-windows-for-
object-detection-with-python-and-opencv/
    '''
    # slide a window across the image
    for y in range(0, image.shape[0], stepSize):
        for x in range(0, image.shape[1], stepSize):
            # yield the current window
            yield (x, y, image[y:y + windowSize[1], x:x + windowSize[0]])

def redefine_ROI(ROI, results, window, charactersType, modelsOriginal,
modelsDCT, modelsDWT, modelsHough, debug=False):
    '''
    Fonction permettant de déterminer de nouvelles zones de recherche une
fois un caractère trouver dans l'image
    Entrée: Image de la région d'intérêt, liste de caractères détectés,
paramètre de la fenêtre, type de charactère attendu,
    modèles des réseaux sur l'image original, DCT, DWT et Hough, mode
déboggage.
    Sortie: Liste de caractères détectés et leur position
    Par: Marc-Antoine Coulombe
    Dernière modification: 2024-03-31
    '''
    characters = []
    characterPos = results[-1][2]
    results = sorted(results, key=lambda x: x[2][0], reverse=False)
    listPos = [item[2] for item in results]
    reservedZones = [[item[0], item[0]+item[2]]for item in listPos]
    #We check if we have enough room on the left of the image for another
window
    if reservedZones[0][0] > 0 and reservedZones[0][0] >= window[0]:
        if debug: print("Checking left part of the image to find another
letter.")
        characters = window_detection(ROI, charactersType, modelsOriginal,
modelsDCT, modelsDWT, modelsHough, results = results, seuil=0.8,
            xstart=0, xend=reservedZones[0][0], debug = debug)
```

```
    else:
        if debug: print("Not enough room at the beginning of the image for a
window.")
        characters = results

    #We check if we have enough room on the right to check for another
window
    newStart = reservedZones[-1][1]
    newROIwidth = ROI.shape[1] - newStart
    if newROIwidth >= window[0]:
        if debug: print("Checking right part of the image to find another
letter.")
        characters = window_detection(ROI, charactersType, modelsOriginal,
modelsDCT, modelsDWT, modelsHough, results = results, seuil=0.8,
            xstart=newStart, xend=ROI.shape[1]-window[0], debug = debug)
    else:
        if debug: print("Not enough room on the right of the detected letter
for a window.")
        characters = results

    #we check if we have enough room between detected letters for windows
    if len(results) > 1:
        for i in range(0, len(results) - 1):
            currentLetterEnd = results[i][2][0]+results[i][2][2]
            NextLetterstart = results[i+1][2][0]
            if NextLetterstart - currentLetterEnd >= window[0]:
                if debug: print(f"Enough room between letter {results[i][0]}
and {results[i+1][0]} for a window")
                characters = window_detection(ROI, charactersType,
modelsOriginal, modelsDCT, modelsDWT, modelsHough, results = results,
seuil=0.8,
                    xstart=currentLetterEnd, xend=NextLetterstart-window[0],
debug = debug)
            else:
               if debug: print(f"Not enough room between letter
{results[i][0]} and {results[i+1][0]} for a window.")
    return characters

def separate_characters(imgROI, charactersPos, firstGuess, charactersType,
modelsOriginal, modelsDCT, modelsDWT, modelsHough, debug=False):
    '''
```

```
    Fonction tentant de séparer une région pouvant contenir plusieurs
caractères ne pouvant être segmentés
    Entrée: Image de la région d'intérêt, position du contour dans l'image,
résultat de la première détection
        type du caractère attendu, modèles de réseaux originaux, DCT, DWT et
Hough, mode deboggage.
    Sortie: Liste de caractères détectés
    Par: Marc-Antoine Coulombe
    Dernière modification: 2024-03-31
    '''
    characters = firstGuess
    imgCharacter = pretraitement.extract_contours(imgROI, charactersPos,
mask=True, padding=False, debug = debug)
    if charactersPos[2] >= 32:
        if debug: print("First guess: ", firstGuess)
        window_starttime = time.time()
        characters = window_detection(imgCharacter, charactersType,
modelsOriginal, modelsDCT, modelsDWT, modelsHough, results=[], seuil=0.95,
xstart=0, xend=0, scale=1, debug = debug)
        window_endtime = time.time()
        elapsedWindow = "{:.2f}".format(window_endtime-window_starttime)
        print(f"Window took: {elapsedWindow} seconds")
        return characters
    else:
        pass
    return characters

def window_detection(image, charactersType, modelsOriginal, modelsDCT,
modelsDWT, modelsHough, winSize = (), results=[], seuil=0.95, xstart=0,
xend=0, scale=1, debug=False):
    '''
    Fonction tentant de détecter des caractères en glissant une fenêtre sur
la région
    Entrée: Image, type de caractère attendu, liste de modèles originaux,
DCT, DWT et Hough,
        Taille de la fenêtre, résultats précédents, seuil de détection de
caractère,
        Position de début du fenêtrage, pôsition de fin, facteur d'échelle
de l'image et déboggage
    Sortie: Liste de caractères détectés
    Par: Marc-Antoine Coulombe
    Dernière modification: 2024-03-31
```

```
    '''
    imageW = int(image.shape[1] * scale)
    imageH = int(image.shape[0] * scale)
    if imageW < 10 or imageH < 10:
        return results
    else:
        image = pretraitement.img_resize_scale(image, scale)
    resultsPyramid = []
    (winW, winH) = (20, image.shape[0])
    if xend == 0:
        xend = image.shape[1]
    sliding_startTime = time.time()
    sliding_limitTime = 10
    # loop over the sliding window for each layer of the pyramid
    for (x, y, window) in sliding_window(image, stepSize=2,
windowSize=(winW, winH)):
        if time.time()-sliding_startTime > sliding_limitTime:
            continue
        # if the window does not meet our desired window size, ignore it
        if window.shape[0] != winH or window.shape[1] != winW:
            continue
        if (x < xstart) or (x >= xend):
            continue

        clone = image.copy()
        cv2.rectangle(clone, (x, y), (x + winW, y + winH), (255), 1)
        window = pretraitement.extract_contours(image, [x, y, winW, winH,
[0]], mask=False, padding=True)
        window = cv2.resize(window, (28, 28))
        imgCharacter, imgCharacterDCT, imgCharacterDWT, imgCharacterHough =
transform.generate_all_transforms(window, debug=False)


        if charactersType == 0:
            charac, characPredValue =
tflite.prediction_numbers(imgCharacter, modelsOriginal, imgCharacterDCT,
modelsDCT, imgCharacterDWT,
                modelsDWT, imgCharacterHough, modelsHough, debug=False)
        elif charactersType == 1:
            charac, characPredValue =
tflite.prediction_letters(imgCharacter, modelsOriginal, imgCharacterDCT,
modelsDCT, imgCharacterDWT,
```

```
                modelsDWT, imgCharacterHough, modelsHough, debug=False)
        else:
            print(bcolors.FAIL + "ERREUR DANS LE TYPE DE CARACTÈRE DÉTECTÉ:
ARRÊT" + bcolors.ENDC)
            exit()

        resultsPyramid.append([charac, characPredValue, (x, y, winW, winH)])
        if debug == True:
            print("Caractère: ", charac)
            print("Predidction: ", characPredValue)
            cv2.imshow("Fenêtre", clone)
            cv2.waitKey(0)
            cv2.destroyAllWindows()

    if len(resultsPyramid)>0:
        resultsPyramid = sorted(resultsPyramid, key=lambda x: x[1],
reverse=True)
        if resultsPyramid[0][1]>= seuil:
            if debug: print(f"Found a high probability character:
{resultsPyramid[0][0]} with weight {resultsPyramid[0][1]}")
            results.append(resultsPyramid[0])
            results = redefine_ROI(image, results, [winW, winH],
charactersType, modelsOriginal, modelsDCT, modelsDWT, modelsHough,
debug=debug)
            results = sorted(results, key=lambda x: x[2][0], reverse=False)
            #break
        else:
            if debug: print(f"Found no character: Scaling image down and
retrying...")
            results = window_detection(image, charactersType,
modelsOriginal, modelsDCT, modelsDWT, modelsHough, winSize = (), results=[],
seuil=0.95, xstart=0, xend=0, scale=2/3, debug=debug)
    if len(results) == 0:
        if debug: print("No character found with high enough certainty")
        results.append(['?', 0, (0, 0, image.shape[1], image.shape[0])])
    return results
```

ANNEXE XI : PROGRAMME DE POST-TRAITEMENT

```
'''
Nom du programme : Programme de post-traitement
Nom du fichier : posttraitement.py
Contenu du fichier : Fonctions pour la détection du type de correction,
comparaison avec un dictionnaire, calcul de la distance Levenshtein et
vérification de la date
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

from asyncio.windows_events import NULL
from datetime import date
import utils
import pretraitement
import transform
import tflite

def levenshteinDistance(A, B):
    '''
    Fonction permettant de calculer la distance Levensthein entre deux
strings
    Entrée: Mot A et mot B
    Sortie: Le nombre de modifications minimal
    Par: https://www.scaler.com/topics/levenshtein-distance-python/
    Dernière modification: 2023-05-04
    '''
    N, M = len(A), len(B)
    # Create an array of size NxM
    dp = [[0 for i in range(M + 1)] for j in range(N + 1)]

    # Base Case: When N = 0
    for j in range(M + 1):
        dp[0][j] = j
    # Base Case: When M = 0
    for i in range(N + 1):
        dp[i][0] = i
    # Transitions
    for i in range(1, N + 1):
        for j in range(1, M + 1):
            if A[i - 1] == B[j - 1]:
                dp[i][j] = dp[i-1][j-1]
```

```
            else:
                dp[i][j] = 1 + min(
                    dp[i-1][j], # Insertion
                    dp[i][j-1], # Deletion
                    dp[i-1][j-1] # Replacement
                )

    return dp[N][M]

def compare_with_database(result, caseContent, debug=False):
    '''
    Fonction permettant de calculer la distance Levensthein entre deux
strings
    Entrée: Mot A et mot B
    Sortie: Le nombre de modifications minimal
    Par: https://www.scaler.com/topics/levenshtein-distance-python/
    Dernière modification: 2023-05-04
    '''
    try:
        rows = utils.read_rows_csv(f"{caseContent}.csv", "\\\\vmaw-dips-
mus2d\\ec_avm_test_m.a.c\\Test\\")
        value = ""
        correction = "NONE"
        nbModifications = 999
        nbModificationsI = 999
        nbModificationsL = 999
        nbModificationsMax = 999
        score = -1
        scoreMax = -999

        resulti = result.replace("L","I")
        resultL = result.replace("I","L")

        for i in range(len(rows)):
            if nbModificationsMax == 0:
                pass
            else:
                row = rows[i].split(",")
                value = row[0]
                nbModifications = levenshteinDistance(result, value)
                nbModificationsI = levenshteinDistance(resulti, value)
                nbModificationsL = levenshteinDistance(resultL, value)
```

```
                if  nbModifications <= nbModificationsI and nbModifications
<= nbModificationsL:
                    if nbModifications < nbModificationsMax:
                        correction = value
                        nbModificationsMax = nbModifications
                        scoreMax = float(row[1])
                    elif nbModifications == nbModificationsMax:
                        score = float(row[1])
                        if score > scoreMax:
                            scoreMax = score
                            correction = value
                            nbModificationsMax = nbModifications
                elif  nbModificationsI < nbModifications and
nbModificationsI < nbModificationsL:
                    if nbModificationsI < nbModificationsMax:
                        correction = value
                        nbModificationsMax = nbModificationsI
                        scoreMax = float(row[1])
                    elif nbModificationsI == nbModificationsMax:
                        score = float(row[1])
                        if score > scoreMax:
                            scoreMax = score
                            correction = value
                            nbModificationsMax = nbModificationsI
                elif  nbModificationsL < nbModifications and
nbModificationsL < nbModificationsI:
                    if nbModificationsL < nbModificationsMax:
                        correction = value
                        nbModificationsMax = nbModificationsL
                        scoreMax = float(row[1])
                    elif nbModificationsL == nbModificationsMax:
                        score = float(row[1])
                        if score > scoreMax:
                            scoreMax = score
                            correction = value
                            nbModificationsMax = nbModificationsL
                else:
                    pass
        if debug:
            print("Detected is: ", result)
            print("Recommendation for correction is: ", correction)
```

```
            print("The minimum number of modifications required is:",
nbModificationsMax)
        return correction, nbModificationsMax
    except:
        print(utils.bcolors.FAIL + "No file detected for correction. Moving
to next case" + utils.bcolors.ENDC)
        return correction

def get_pos_nums(num):
    '''
    Fonction permettant d'obtenir la date dans une liste afin d'accéder à
chacun des éléments individuellement
    Entrée: Nombre à réécrire en liste
    Sortie: Liste de chacun des chiffres composant le nombre
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-05-04
    '''
    pos_nums = []
    while num != 0:
        pos_nums.insert(0, num % 10)
        num = num // 10
    return pos_nums

def validate_date(OCRText, expectedContent, debug=False):
    '''
    Fonction permettant de réaliser plusieurs tests sur la date afin de
s'assurer de sa validité
    Entrée: Date obtenu en format string, contenu attendue dans la case et
mode deboggage
    Sortie: Correction suggérée par le système de reconnaissance
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-05-04
    '''
    try:
        OCRDate = int(OCRText)
        if expectedContent == "year_of_birth":
            if (1900 <= OCRDate <= date.today().year):
                return OCRDate
            else:
                date_number = get_pos_nums(OCRDate)
                if len(date_number) == 4:
                    if date_number[0] == 1:
```

```
                        #Nécessairement, si l'on se trouve dans le premier
millénnaire, on se trouve au 20e siècle
                        date_number[1] = 9
                        s = [str(i) for i in date_number]
                        OCRDate = str("".join(s))
                        return OCRDate
                    elif  date_number[1] == 9:
                        date_number[0] = 1
                        s = [str(i) for i in date_number]
                        OCRDate = str("".join(s))
                        return OCRDate
                print(utils.bcolors.FAIL + "Year of birth is impossible.
Please ensure no mistake has been made" + utils.bcolors.ENDC)
                return "IMPOSSIBLE VALUE"

        elif expectedContent == "month_of_birth":
            if (1 <= OCRDate < 13):
                return OCRText
            else:
                print(utils.bcolors.FAIL + "Month of birth is impossible.
Please ensure no mistake has been made" + utils.bcolors.ENDC)
                return "IMPOSSIBLE VALUE"
        elif expectedContent == "day_of_birth":
            if (1 <= OCRDate < 32):
                return OCRText
            else:
                print(utils.bcolors.FAIL + "Day of birth is impossible.
Please ensure no mistake has been made" + utils.bcolors.ENDC)
                return "IMPOSSIBLE VALUE"
    except:
        correction = "IMPOSSIBLE VALUE"
        print(utils.bcolors.FAIL + "ERROR: Can't convert Date to int." +
utils.bcolors.ENDC)


def correction_by_case(OCRText, correctionType, expectedContent,
streetNumber, imgROI, country, modelsLettres, debug = False):
    '''
    Fonction permettant de réaliser la correction appropriée selon le type
de correction sélectionné.
    Entrée: Texte à corriger, type de correction attendu, contenu attendu
dans la case, Numéro de rue (dans le cas d'une vérification d'adresse),
```

```
        image de la case, pays, modèles de réseaux pour les lettres et mode
deboggage.
    Sortie: Liste de chacun des chiffres composant le nombre
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-05-04
    '''
    correction = "NONE"
    match correctionType:
        case 0: #No correction
            correction = OCRText
        case 1: #Using database for correction
            correction = compare_with_database(OCRText, expectedContent,
debug=debug)
        case 2: #Date of birth
            correction = validate_date(OCRText, expectedContent,
debug=debug)
        case 3: #Address
            streetName = OCRText[len(streetNumber):]
            if expectedContent == "number_and_street" or expectedContent ==
"mailing_address" or expectedContent == "mailing_address_unit":
                for i in range(len(streetName)):
                    if i>0 and i<len(streetName)-1:
                        if ord(streetName[i-1]) > 58 and
ord(streetName[i+1]) > 58 and ord(streetName[i]) < 58:
                            charactersPos =
pretraitement.contours_ROI(imgROI, debug = debug)
                            imgCharacter =
pretraitement.extract_contours(imgROI, charactersPos[i+len(streetNumber)],
padding=True, debug = debug)
                            imgCharacter, imgCharacterDCT, imgCharacterDWT,
imgCharacterHough = transform.generate_all_transforms(imgCharacter, debug =
debug)
                            charac, characPredValue =
tflite.prediction_letters(imgCharacter, modelsLettres[0], imgCharacterDCT,
modelsLettres[1], imgCharacterDWT,
                                modelsLettres[2], imgCharacterHough,
modelsLettres[3])
                            streetName = list(streetName)
                            streetName[i] = charac
                            streetName = utils.chars_to_string(streetName)
                correction = streetNumber + " " + streetName
```

```
            elif ("province" in expectedContent or "city" in
expectedContent) and (country == 'CANADA'):
                if debug:
                    print("Since mailing or unit address is in Canada, using
dictionnary for validation")
                correction = compare_with_database(OCRText, expectedContent,
debug=debug)
            else:
                correction = OCRText


        case 4:
            correction = OCRText
    return correction
```

ANNEXE XII : PROGRAMME GÉNÉRANT LES FICHIERS DE SORTIES

```
'''
Nom du programme : Programme générant les fichiers de sorties
Nom du fichier : output.py
Contenu du fichier : Fonctions générant le fichier de sorti en format .csv
compilant les résultats
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import utils
import os

def generate_temp_header(pathProjetTest):
    '''
    Fonction permettant d'écrire l'entête au fichier désginé
    Entrée: Chemin d'accès du fichier
    Sortie: Écriture dans le ficher CSV
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    headerTemp= ["Contenu", "Type de caractères (Chiffre: 0, lettres: 1,
Mixte:2, 3:Alternance Lettres/Chiffres, Bool:4)", "Texte reconnu",
    "Temps requis reconnaissance (s)", "Type de correction (0: Aucune, 1:
Dictionnaire, 2: Date, 3: Adresse", "Correction suggérée", "Temps requis
correction (s)"]
    export_to_temp(headerTemp, pathProjetTest)
    return 0

def export_to_temp(content, path):
    '''
    Fonction permettant d'écrire une ligne au fichier temporaire
    Entrée: Liste du contenu de la ligne, chemin d'accès vers le fichier CSV
temporaire
    Sortie: Écriture dans le ficher CSV
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    filename = "tempfile.csv"
    utils.write_row_csv(content, filename, path)
    return 0
```

```
def delete_temp(path):
    '''
    Fonction permettant de générer un fichier de sorti après la
reconnaissance d'un formulaire
    Entrée: Nom du fichier, chemin d'accès vers le dossier de sauvegarde et
debug
    Sortie: Écriture du fichier sur le disque
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    filename = "tempfile.csv"
    file = os.path.join(path, filename)
    try:
        if(os.path.exists(file) and os.path.isfile(file)):
            os.remove(file)
            print("Detection completed, deleting temporary files...")
    except:
        print(utils.bcolors.FAIL + "Error: couldn't remove temporary file" +
utils.bcolors.ENDC)
    return 0


def generate_output_file(outputName, path, debug=False):
    '''
    Fonction permettant de générer un fichier de sorti après la
reconnaissance d'un formulaire
    Entrée: Nom du fichier, chemin d'accès vers le dossier de sauvegarde et
debug
    Sortie: Écriture du fichier sur le disque
    Par: Marc-Antoine Coulombe
    Date: 2023-10-31
    '''
    tempName = "tempfile.csv"
    rows = utils.read_rows_csv(tempName, path)
    utils.write_row_csv(rows, outputName, path)
    if not debug:
        delete_temp(path)

    return 0
```

ANNEXE XIII : PROGRAMME DE FONCTIONS UTILITAIRES

```
'''
Nom du programme : Programme de fonctions utilitaires
Nom du fichier : utils.py
Contenu du fichier : Fonctions générales n'ayant pas de catégorie propre
Nom du programmeur : Marc-Antoine Coulombe
Date : 2024-03-19
Numéro de version : 1.0
'''

import numpy as np
from keras.utils import Sequence
import tensorflow as tf
import time
from watchdog.observers import Observer
from watchdog.events import FileSystemEventHandler
import os
import csv

def normalizeuint8(I):
    '''
    Fonction permettant de normaliser une image en format uint8
    Entrée: Image à normaliser
    Sortie: Retourne l'image normalisée en format np.uint8
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    if I.max() != 0:
        mn = I.min()
        mx = I.max()
        mx -= mn
        I = ((I - mn)/mx) * 255
    else:
        print(bcolors.WARNING + "Image is empty, not normalizing to not
divide by 0" + bcolors.ENDC)
    return I.astype(np.uint8)

def normalizeuint32(I):
    '''
    Fonction permettant de normaliser une image en format uint32
    Entrée: Image à normaliser
    Sortie: Retourne l'image normalisée en format np.uint32
    Par: Marc-Antoine Coulombe
```

```
    Dernière modification: 2023-10-04
    '''
    if I.max() != 0:
        mn = I.min()
        mx = I.max()
        mx -= mn
        I = ((I - mn)/mx) * 4294967295
    else:
        print(bcolors.WARNING + "Image is empty, not normalizing to not
divide by 0" + bcolors.ENDC)
    return I.astype(np.uint32)

def normalizefloat16(I):
    '''
    Fonction permettant de normaliser une image en format float16
    Entrée: Image à normaliser
    Sortie: Retourne l'image normalisée en format np.float16
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    if I.max() != 0:
        mn = I.min()
        mx = I.max()
        mx -= mn
        I = ((I - mn)/mx)
    else:
        print(bcolors.WARNING + "Image is empty, not normalizing to not
divide by 0" + bcolors.ENDC)
    return I.astype(np.float16)

def normalizefloat32(I):
    '''
    Fonction permettant de normaliser une image en format float32
    Entrée: Image à normaliser
    Sortie: Retourne l'image normalisée en format np.float32
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    if I.max() != 0:
        mn = I.min()
        mx = I.max()
        mx -= mn
```

```
        I = ((I - mn)/mx)
    else:
        print(bcolors.WARNING + "Image is empty, not normalizing to not
divide by 0" + bcolors.ENDC)
    return I.astype(np.float32)

def GPU_available():
    '''
    Fonction permettant de vérifier le GPU disponible pour le calcul
    Entrée: NONE
    Sortie: Message console retournant le nom du GPU et la version de
Tensorflow installée
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    print("Num GPUs Available: ",
len(tf.config.list_physical_devices('GPU')))
    tf.test.gpu_device_name()
    print(tf.__version__)
    return None

class DataGenerator(Sequence):
    '''
    Classe permettant de générer un "groupement" pour réduire la mémoire
requise lors de l'apprentissage de réseaux de neurones
    Entrée: NONE
    Sortie: NONE
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    def __init__(self, x_set, y_set, batch_size):
        self.x, self.y = x_set, y_set
        self.batch_size = batch_size

    def __len__(self):
        return int(np.ceil(len(self.x) / float(self.batch_size)))

    def __getitem__(self, idx):
        batch_x = self.x[idx * self.batch_size:(idx + 1) * self.batch_size]
        batch_y = self.y[idx * self.batch_size:(idx + 1) * self.batch_size]
        return batch_x, batch_y
```

```
class bcolors:
    '''
    Classe permettant d'attribuer une couleur au texte dans la console
    Entrée: NONE
    Sortie: NONE
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    Example: print(bcolors.WARNING + "Warning: No active frommets remain.
Continue?" + bcolors.ENDC)
    '''
    HEADER = '\033[95m'
    OKBLUE = '\033[94m'
    OKCYAN = '\033[96m'
    OKGREEN = '\033[92m'
    WARNING = '\033[93m'
    FAIL = '\033[91m'
    ENDC = '\033[0m'
    BOLD = '\033[1m'
    UNDERLINE = '\033[4m'


class DirectoryHandler(FileSystemEventHandler):
    '''
    Classe permettant de définir l'action à effectué selon la modification
détectée dans le répertoire
    Entrée: NONE
    Sortie: NONE
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-04
    '''
    def on_any_event(self, event):
        pass

    def on_created(self, event):
        print("on_created", event.src_path)
        print(event.src_path.strip())
        filename = event.src_path.strip().split("\\")[5]
        print(filename)
        if((event.src_path).strip() == "./Nouveau Document texte.txt" ):
            print("Do something")

def activate_directory_handler(pathProjetTest):
```

```
    '''
    Fonction permettant de surveiller un répertoire (flag)
    Entrée: Chemin d'accès vers le répertoire à surveiller
    Sortie: Aucune
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    event_handler = DirectoryHandler()
    observer = Observer()
    observer.schedule(event_handler, path = pathProjetTest, recursive=False)
    observer.start()
    start_time = time.time()
    i = 0
    while True:
        try:
            end_time = time.time()
            elapsed = end_time - start_time
            if not elapsed % 30:
                print(f'Time elapsed observing file: {elapsed:.2f} seconds')
            i+=1
        except KeyboardInterrupt:
            print('Keyboard interrupt detected, stopping watchdog')
            observer.stop()
            exit()
    return 0

def flatten(A):
    '''
    Fonction permettant de réaliser une liste simple avec une liste de
listes
    Entrée: Liste
    Sortie: Aucune
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    rt = []
    for i in A:
        if isinstance(i,list): rt.extend(flatten(i))
        else: rt.append(i)
    return rt

def chars_to_string(characs):
```

```
    '''
    Fonction permettant de convertir une série de caractère en string
    Entrée: Liste de caractères
    Sortie: String
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    characs = flatten(characs)
    result = ""
    for charac in characs:
      result += charac
    return result

def write_row_csv(content, filename, path):
    '''
    Fonction permettant d'écrire les lignes dans un fichier CSV
    Entrée: Ligne contenant les données désirées, nom du fichier CSV, chemin
d'accès vers le répertoire
    Sortie: Écrture dans le fichier CSV
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    with open(os.path.join(path, filename), 'a', newline='') as file:
        writer = csv.writer(file)
        writer.writerow(content)
    return 0

def read_rows_csv(filename, path):
    '''
    Fonction permettant de lire les lignes dans un fichier CSV
    Entrée: Nom du fichier CSV, chemin d'accès vers le répertoire
    Sortie: Liste des lignes du fichier
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    with open(os.path.join(path, filename), 'r') as file_obj:
        rows = []
        reader = csv.reader(file_obj)
        for row in file_obj:
            rows.append(row.split('\n')[0])
    return rows
```

```
def replace_filename():

    '''
    Fonction permettant de renommer l'ensemble des fichiers dans un
répertoire
    Entrée: Aucune
    Sortie: Fichiers renommés
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    folder = "D:\\Maitrise\\Bases de donnees 2\\Bases de donnees Mixte\\Base
de donnees Mixte Originale Fusion tiff\\"
    pathiter = (os.path.join(root, filename)
        for root, _, filenames in os.walk(folder)
        for filename in filenames
    )

    compteur = 0
    for path in pathiter:
        newname =  path.replace('999.tiff', '1.tiff')
        if newname != path:
            os.rename(path,newname)
        compteur += 1
        if not compteur % 10000:
            print("Renaming image: ", compteur)


def extract_element(lst, element):
    '''
    Fonction permettant d'extraire un élément d'une liste
    Entrée: Liste, élément à extraire
    Sortie: Élément extrait
    Par: Marc-Antoine Coulombe
    Dernière modification: 2023-10-31
    '''
    return[item[element] for item in lst]
```

RÉFÉRENCES BIBLIOGRAPHIQUES

Ahlawat, S., & Choudhary, A. (2020). Hybrid CNN-SVM classifier for handwritten digit recognition. *Procedia computer science, 167,* 2554-2560. doi:10.1016/j.procs.2020.03.309

Ahmed, N., Natarajan, T., & Rao, K. R. (1974). Discrete cosine transform. *IEEE transactions on computers. Institute of Electrical and Electronics Engineers, C-23*(1), 90-93. doi:10.1109/t-c.1974.223784

Albahli, S., Nawaz, M., Javed, A., & Irtaza, A. (2021). An improved faster-RCNN model for handwritten character recognition. *Arabian journal for science and engineering, 46*(9), 8509-8523. doi:10.1007/s13369-021-05471-4

Aqab, S., & Usman, M. (2020). Handwriting recognition using artificial intelligence neural network and image processing. *International journal of advanced computer science and applications : IJACSA, 11*(7). doi:10.14569/ijacsa.2020.0110719

Arivazhagan, S., Arun, M., & Rathina, D. (2021). Recognition of handwritten characters using deep convolution neural network. *Journal of the National Science Foundation of Sri Lanka, 49*(4), 503. doi:10.4038/jnsfsr.v49i4.9825

Baviskar, D., Ahirrao, S., Potdar, V., & Kotecha, K. (2021). Efficient automated processing of the unstructured documents using artificial intelligence: A systematic literature review and future directions. *IEEE access: practical innovations, open solutions, 9,* 72894-72936. doi:10.1109/access.2021.3072900

Bernsen, J. (1986). Dynamic Thresholding of Gray Level Image. *Proceedings of International Conference on Pattern Recognition*, (pp. 1251-1255). Paris.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* New York: Springer New York.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Cheriet, M., Kharma, N., Liu, C.-L., & Suen, C. (2007). *Character recognition systems: A guide for students and practitioners.* New York: Wiley-Interscience.

Chollet, F. (2022). *Deep learning with python.* New York, NY: Manning Publications.

Cilia, N. D., De Stefano, C., Fontanella, F., & Scotto di Freca, A. (2019). A ranking-based feature selection approach for handwritten character recognition. *Pattern recognition letters, 121*, 77-86. doi:10.1016/j.patrec.2018.04.007

Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters. Retrieved from https://arxiv.org/abs/1702.05373

Damerval, C. (2008). Ondelettes pour la détection de caractéristiques en traitement d'images - Applications à la détection de régions d'intérêts. Grenoble, France: Université Joseph Fourier.

Das, D., Nayak, D. R., Dash, R., & Majhi, B. (2019). An empirical evaluation of extreme learning machine: application to handwritten character recognition. *Multimedia tools and applications, 78*(14), 19495-19523. doi:10.1007/s11042-019-7330-0

Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM, 15*(1), 11-15. doi:10.1145/361237.361242

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* London, England: MIT Press.

Gupta, M. R., Jacobson, N. P., & Garcia, E. K. (2007). OCR binarization and image pre-processing for searching historical documents. *Pattern recognition, 40*(2), 389-397. doi:10.1016/j.patcog.2006.04.043

Haar, A. (1910). Zur Theorie der orthogonalen Funktionensysteme: Erste Mitteilung. *Mathematische annalen, 69*(3), 331-371. doi:10.1007/bf01456326

Hough, P. V. (1960). *United States Patent No. US3069654A.*

Hyperscience. (2023, September 25). *Hyperscience*. Retrieved from Hyperscience: https://hyperscience.com/

IBM. (2023, September 25). *IBM Datacap*. Retrieved from IBM: https://www.ibm.com/products/data-capture-and-imaging

Ingle, R. R., Fujii, Y., Deselaers, T., Baccash, J., & Popat, A. C. (2019). A Scalable Handwritten Text Recognition System. *2019 International Conference on Document Analysis and Recognition (ICDAR).* doi:10.1109/icdar.2019.00013

Jyotsna, Chauhan, S., Sharma, E., & Doegar, A. (2016). Binarization techniques for degraded document images — A review. *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO).* doi:10.1109/icrito.2016.7784945

Kaehler, A., & Bradski, G. (2016). *Learning opencv 3: Computer vision in C++ with the opencv library.* O'Reilly Media.

Kavallieratou, E., Likforman-Sulem, L., & Vasilopoulos, N. (2018). Slant removal technique for historical document images. *Journal of imaging, 4*(6), 80. doi:10.3390/jimaging4060080

Khan, S., Rahmani, H., Shah, S. A., & Bennamoun, M. (2018). *A Guide to Convolutional Neural Networks for Computer Vision.* San Rafael, CA, USA: Morgan & Claypool.

Kinsley, H., & Kukieła, D. (2020). *Neural Networks from Scratch in Python.*

Lee R., G., Gommers, R., Wasilewski, F., & Wohlfahrt Kai, O. A. (2019). PyWavelets: A Python package for wavelet analysis. *Journal of Open Source Software, 4*(36), 1237. doi:10.21105/joss.01237

Lukasik, E., Charytanowicz, M., Milosz, M., Tokovarov, M., Kaczorowska, M., Czerwinski, D., & Zientarski, T. (2021). Recognition of handwritten Latin characters with diacritics using CNN. *Bulletin of the Polish Academy of Sciences Technical Sciences, 69*(1), e136210.

Maekawa, K., Tomiura, Y., Fukuda, S., Ishita, E., & Uchiyama, H. (2019). Improving OCR for historical documents by modeling image distortion. *Digital Libraries at the Crossroads of Digital Information for the Future*, 312-316. doi:10.1007/978-3-030-34058-2_31

Martínek, J., Lenc, L., & Král, P. (2020). Building an efficient OCR system for historical documents with little training data. *Neural computing & applications*, 17209-17227. doi:10.1007/s00521-020-04910-x

Mekouar, M. (2001). Compression d'images médicales par ondelettes et régions d'intérêt. Montréal: École de techologie supérieure - Université du Québec.

Memon, J., Sami, M., Khan, R. A., & Uddin, M. (2020). Handwritten optical character recognition (OCR): A comprehensive systematic literature review (SLR). *IEEE access: practical innovations, open solutions, 8*, 142642-142668. doi:10.1109/access.2020.3012542

Mont-Royal, L. s. (2023). *Bibliothèque et Archives Canada*. Récupéré sur Ville de Montréal: https://ville.montreal.qc.ca/siteofficieldumontroyal/patrimoine-documentaire-archivistique/bibliotheque-archives-canada-bac#:~:text=La%20section%20%C2%AB%20Th%C3%A8ses%20Canada%20%C2%BB%20permet,sont%20consultables%20sous%20forme%20num%C3%A9rique).

Nguyen, Q.-D., Le, D.-A., Phan, N.-M., & Zelinka, I. (2021). OCR error correction using correction patterns and self-organizing migrating algorithm. *Pattern analysis and applications: PAA, 24*(2), 701-721. doi:10.1007/s10044-020-00936-y

Nguyen, T.-T.-H., Jatowt, A., Coustaty, M., Nguyen, N.-V., & Doucet, A. (2019). Deep statistical analysis of OCR errors for effective post-OCR processing. *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL).* doi:10.1109/jcdl.2019.00015

Niblack, W. (1986). *An introduction to digital image processing.* Prentice Hall.

Numpy. (2023, September 27). *numpy.fft.fft2*. Retrieved from Numpy: https://numpy.org/doc/stable/reference/generated/numpy.fft.fft2.html

OpenCV. (2023, September 26). *Basic concepts of the homography explained with code*. Retrieved from OpenCV: https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html

OpenCV. (2023, September 26). *Hough Line Transform*. Retrieved from OpenCV: https://docs.opencv.org/4.x/d6/d10/tutorial_py_houghlines.html

OpenCV. (2023, September 26). *Image Thresholding*. Retrieved from OpenCV: https://docs.opencv.org/4.x/index.html

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics, 9*(1), 62-66. doi:10.1109/tsmc.1979.4310076

Pandian, K., Mathivanan, P., & Ganesamoorthy, B. (2011). Analysis of handwritten image using feature extraction algorithm of texture images. Dans *International Conference on Computer Technology and Development, 3rd (ICCTD 2011)* (pp. 467-471). New York: ASME Press.

Parida, P., & Bhoi, N. (2017). Wavelet based transition region extraction for image segmentation. *Future Computing and Informatics Journal, 2*(2), 65-78. doi:10.1016/j.fcij.2017.10.005

Povolotskiy, M., & Tropin, D. (2019). Dynamic programming approach to template-based OCR. *Eleventh International Conference on Machine Vision (ICMV 2018).* SPIE. doi:10.1117/12.2522974

Rabhi, B., Elbaati, A., Hamdi, Y., & Alimi, A. M. (2019). Handwriting recognition based on temporal order restored by the end-to-end system. *2019 International Conference on Document Analysis and Recognition (ICDAR).* doi:10.1109/icdar.2019.00199

Rosebrock, A., Thanki, A., Paul, S., & Haase, J. (2020). *OCR with OpenCV, Tesseract, and Python - Intro to OCR.*

Rosyda, S. S., & Purboyo, T. W. (2018). A Review of Various Handwriting Recognition Methods. *International Journal of Applied Engineering Research, 13*(2), 1155-1164.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *2011 International Conference on Computer Vision.* doi:10.1109/iccv.2011.6126544

Sauvola, J., & Pietikäinen, M. (2000). Adaptive document image binarization. *Pattern recognition, 33*(2), 225-236. doi:10.1016/s0031-3203(99)00055-2

Szeliski, R. (2023). *Computer vision: Algorithms and applications* (2nd ed.). Cham, Switzerland: Springer Nature.

Tensorflow. (2023, 07 06). *Module: tf.keras.activations.* Récupéré sur Tensorflow: https://www.tensorflow.org/api_docs/python/tf/keras/activations

Tesseract-OCR. (2023, September 25). *Tesseract User Manual*. Retrieved from Tesseract-OCR: https://tesseract-ocr.github.io/tessdoc/

Transfer learning for handwriting recognition on historical documents. (2018). *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods.* SCITEPRESS - Science and Technology Publications. doi:10.5220/0006598804320439

Wigington, C., Tensmeyer, C., Davis, B., Barrett, W., Price, B., & Cohen, S. (2018). Start, follow, read: End-to-end full-page handwriting recognition. (S. I. Publishing, Éd.) *Computer Vision – ECCV 2018*, 372-388. doi:10.1007/978-3-030-01231-1_23

Xue, Y., Tong, Y., Yuan, Z., Su, S., Slowik, A., & Toglaw, S. (2021). Handwritten character recognition based on improved convolutional neural network. *Intelligent automation & soft computing, 29*(2), 497-509. doi:10.32604/iasc.2021.016884

Yousef, M., Hussain, K. F., & Mohammed, U. S. (2020). Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. *Pattern recognition, 108*(107482), 107482. doi:10.1016/j.patcog.2020.107482

Zhang, T. Y., & Suen, C. Y. (1984). A fast parallel algorithm for thinning digital patterns. *Communications of the ACM, 27*(3), 236-239. doi:10.1145/357994.358023