# Apprentissage profond auto-supervisé de métriques : application à la prédiction d'assemblage de fragments de papyrus

Antoine Pirrone

# THÈSE

présentée à

# L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

par


# ANTOINE PIRRONE


POUR OBTENIR LE GRADE DE

# DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

---

# Apprentissage profond auto-supervisé de métriques : application à la prédiction d'assemblage de fragments de papyrus

---

**Date de soutenance :** 28 Mars 2022

**Devant la commission d'examen composée de :**

| | | | | |
|---|---|---|---|---|
| Harold | Mouchère | Prof. | Université de Nantes | Président |
| Véronique | Eglin | Prof. | INSA Lyon | Rapporteuse |
| Andreas | Fischer | Prof. | HES-SO | Rapporteur |
| Marie-Pierre | Chaufray | CR | CNRS, Ausonius | Examinatrice |
| Daniel | Stökl Ben Ezra | DE | EPHE | Examinateur |
| Marie | Beurton-Aimar | MCF-HDR | Université de Bordeaux | Directrice |
| Nicholas | Journet | MCF-HDR | Université de Bordeaux | Directeur |

– 2021 –

**Titre**

Apprentissage profond auto-supervisé de métriques : application à la prédiction d'assemblage de fragments de papyrus

**Résumé**

Dans cette thèse est présenté le développement de méthodes d'apprentissage profond pour proposer des appairages pertinents de fragments de documents, appliquées à la reconstruction de papyrus anciens. Nous avons conçu une méthode basée sur des réseaux de neurones siamois profonds ainsi qu'une approche *auto-supervisée* pour pallier le problème de la faible quantité de données d'entraînement annotées disponibles dans ce domaine.

Ce travail fait partie du projet *GESHAEM*, une initiative de recherche scientifique financée par le Conseil Européen de la Recherche (European Research Council, ERC). Son objectif est d'étudier un corpus papyrologique, la collection *Jouguet* de la Sorbonne, composé de papyrus qui ont été déchirés pour fabriquer des ornements funéraires. Ces papyrus sont pour la plupart des documents administratifs et fiscaux dont le contenu est très utile pour les historiens afin de mieux comprendre la façon dont ces sociétés anciennes étaient organisées. Cependant, de part le fait que les papyrus ont été déchirés il y a environ 2000 ans, et qu'ils ont été très dégradés à cause du temps et des conditions de conservation, leur étude nécessite un grand travail de reconstruction par les papyrologues. C'est un processus long et fastidieux s'il est fait à la main, car la collection est composée de centaines de fragments.

L'objectif de ce travail est ainsi de calculer des suggestions d'appairages de fragments aux papyrologues afin d'accélérer le processus de reconstruction. Nous proposons une approche basée sur des patchs utilisant des réseaux de neurones siamois profonds pour calculer un score de similarité entre deux fragments de papyrus. La difficulté d'obtenir des données annotées en grande quantité rend difficile la construction d'un modèle fonctionnant sur différents types de documents. Pour cela, nous proposons une méthodologie d'apprentissage *auto-supervisée* exploitant la structure intrinsèque des documents. Cette méthode permet d'entraîner des réseaux de neurones siamois profonds sans avoir besoin de données annotées. Nous avons aussi constitué une base de données prête à l'emploi basée sur la collection de papyrus de l'Université du Michigan, que nous avons ouverte à la communauté.

**Mots-clés**

Apprentissage automatique, Analyse et traitement d'image, Apprentissage profond, Assemblage automatisé, Archéologie, Papyrologie

**Title**

Self-supervised Deep Metric Learning applied to papyrus reconstruction

**Abstract**

This work is about developing Deep Learning methods to suggest relevant pairings for fragments of documents applied to the reconstruction of ancient papyrus fragments. A method based on Deep Siamese Neural Networks and a self-supervised approach was developed to tackle the challenge of the scarcity of training data.

This work is part of the *GESHAEM* project, a scientific research initiative funded by the European Research Council (ERC). Its goal is to study a papyrological corpus, the *Jouguet* Collection of the Sorbonne, composed of papyri that have been torn to make funeral ornaments (cartonnages). These papyri are mostly administrative and fiscal documents that are very interesting to historians in order to get a better understanding of the way these ancient societies were organized. However, because the papyri were torn at the time (about 2000 years ago), and because they are quite degraded as a result of the elements and time, their study first requires a lot of reconstruction work by papyrologists. This is a long and tedious process to do manually as the collection is composed of hundreds of fragments.

The goal of this work is therefore to automatically propose suggestions of pairings of fragments to the papyrologists in order to speed up the reconstruction process. A patch based approach using Deep Siamese Neural Networks was developed to compute a similarity score between two fragments of papyrus. This is a challenging task for multiple reasons as in this field data is often scarce, not (yet) labeled and in poor condition. Moreover, it is very difficult to get enough labeled data to produce a general model that will work on many types of documents. The proposed process should be general enough to provide good results on any kind of documents (or even other applicative fields) by training domain specific models.

To tackle these issues, a self-supervised learning process that uses the intrinsic structure of the documents was developed to allow for the training of Deep Siamese Neural Networks without the need for any labeled data. A ready to use learning database based on the papyrological collection of the University of Michigan was also proposed as a contribution to the community.

**Keywords**

Machine learning, Image processing, Deep learning, Information retrieval, Archeology, Papyrology

# Remerciements

Je tiens dans un premier temps à remercier chaleureusement mes deux directeurs de thèse, Nicholas Journet et Marie Beurton-Aimar, pour la qualité de leur encadrement ainsi que pour leur soutien indéfectible, sans quoi cette thèse aurait sans doute été bien plus difficilement menée à son terme. Je sais ma chance d'avoir pu travailler avec des encadrants aussi présents et bienveillants.

Je voudrais ensuite remercier les membres de mon jury de thèse. Un grand merci à Véronique Eglin et à Andreas Fischer d'avoir accepté d'être rapporteurs de ma thèse, merci aussi à Harold Mouchère, Daniel Stoekl Ben Ezra et Marie-Pierre Chaufray d'avoir accepté d'être examinateurs lors de ma soutenance.

Je souhaite aussi remercier les membres du projet *GESHAEM* avec qui j'ai eu la chance d'échanger et de travailler au cours de cette thèse. Merci à Florent Jacques, Lorenzo Uggetti et Adam Bülow-Jacobsen pour leur accueil chaleureux lors de ma visite de la collection à la Sorbonne en début de thèse. Merci à Nathalie Prévôt, avec qui il a été très agréable d'encadrer des stages, et d'échanger plus généralement sur des questions techniques liées au projet. Enfin, un merci tout particulier à Marie-Pierre Chaufray, qui a été d'une aide précieuse tout au long de ma thèse, et plus particulièrement lors de la rédaction de ce manuscrit en répondant à mes nombreuses questions concernant la papyrologie et les détails de la collection Jouguet.

Mes remerciements vont aussi à tous mes collègues du LaBRI et plus particulièrement à notre petit groupe de travail du jeudi pour ces discussions scientifiques toujours intéressantes. Merci à Cécil, Myriam, Linh, Tú, Kevin, Charlotte et Benoît.

Je souhaite également remercier tous les membres, anciens et actuels, de l'équipe Rhoban qui m'accompagnent dans cette aventure académique depuis déjà 6 ans à travers leurs conseils et leurs expériences. Nos discussions ont beaucoup aidé à me forger une idée réaliste de ce qu'est la recherche et ont contribué à ma décision de me lancer dans une thèse. Je suis très heureux d'avoir pu partager avec cette équipe de grands moments de travail acharné en préparation, ainsi que pendant la RoboCup. Un immense merci à Steve, Olivier, Grégoire, Ludovic, Loïc, Patxi, Rémi, Quentin, Julien, Adrien, Lucie et Antun.

Je voudrais aussi remercier toute ma famille pour leurs encouragements constants et plus particulièrement mes parents, qui m'ont soutenu durant toutes ces longues années d'études. Je suis très heureux d'avoir une grande famille aussi soudée et bienveillante, un grand merci à vous tous.

Merci à tous mes amis, pour la plupart de très longue date. Je suis extrêmement heureux de faire partie de ce grand groupe d'amis aux intérêts et aux parcours divers, mais qui reste soudé avec les années, et parfois malgré la distance. Un grand merci

à Alexandre, Antoine, Maxime, Quentin, Batst, Pierje, Zazar, Élodie, GG, Nicho, Christophe, Chloé, Thomas, Marine, Clément, Mélodie, Cyril, Elsa, Pierrot, Anne, Dylan, Caroline, Jérôme, Simon et Grégoire.

Enfin, je souhaite remercier ma compagne, Alyssa, qui a vécu ma thèse à mes côtés, qui était toujours présente pour me soutenir, autant dans les périodes de joies que les passages difficiles. Merci pour tout.

# Sommaire

# Introduction

Les documents anciens sont des ressources extrêmement précieuses pour les historiens travaillant à comprendre le fonctionnement d'anciennes civilisations. Ils peuvent contenir des informations sur l'organisation administrative, politique, sociale ou religieuse de ces sociétés anciennes. Au cours de l'histoire, de nombreux types de supports d'écriture ont été utilisés dans différentes régions du monde et ensuite retrouvés par les archéologues. On peut par exemple citer les tablettes d'argile, les stèles de pierre, les ostraca, le parchemin, le papier ou encore le papyrus.

Parmi ces supports, nous nous intéressons ici particulièrement au papyrus. Ce support d'écriture fut très utilisé en Égypte et autour de la méditerranée durant des milliers d'années. Depuis des décennies, des fouilles archéologiques ont permis de retrouver des papyrus datant de différentes époques. Il n'est pas rare que ces papyrus, en plus d'être très dégradés par le temps et les conditions de conservation, soient retrouvés sous la forme de fragments. Il est donc nécessaire pour les papyrologues de reconstruire le document original pour pouvoir l'étudier.

Ce processus de reconstruction des papyrus est aujourd'hui très long et difficile, car il est fait manuellement par les papyrologues. Ce travail est fait directement en manipulant les fragments, ou sur ordinateur à partir de leurs versions numérisées. La numérisation des collections de papyrus est de plus en plus répandue, elle est ainsi l'occasion de développer des méthodes automatiques pour aider les papyrologues dans leur travail. Nous distinguons deux étapes menant à la reconstruction de ces documents. La première est celle du tri des fragments, dans laquelle sont déterminés quels fragments qui font partie du même document. La seconde est celle de l'assemblage lui-même, dans laquelle les fragments sont positionnés correctement pour reconstruire le document.

**Cette thèse se concentre sur la première étape, l'objectif est de proposer des suggestion d'appairage de fragments aux papyrologues pour accélérer le processus de reconstruction**. Notre méthode se base sur des modèles d'apprentissage profond utilisant des architectures de réseaux de neurones appelés *réseaux siamois* pour apprendre à calculer des scores de similarité entre des images de documents. Nous proposons une méthodologie d'entraînement *auto-supervisée* ne nécessitant pas d'avoir recours à des données annotées, qui sont souvent rares et difficiles à obtenir dans le domaine de l'analyse de documents anciens.

L'approche proposée a été évaluée en profondeur sur deux jeux de données de documents internationaux et publiques. En plus de cela, nous avons étudié la pertinence des appairages de fragments proposés par notre approche sur les papyrus

de la collection *Jouguet*. Ces tests ont été réalisés en collaboration avec les papyrologues du projet *GESHAEM*. Un logiciel appelé *Orio* a aussi été développé pour permettre aux papyrologues de manipuler les fragments numérisés et d'accéder aux propositions d'appairages de notre modèle.

## Organisation du manuscrit

Dans le **Chapitre** 1, nous définissons ce qu'est la *Papyrologie* et la *Papyrologie Numérique* et discutons des problématiques importantes de ces domaines. Nous dressons ensuite un état des lieux des différentes spécialités qui composent le domaine de recherche qu'est la *Papyrologie Numérique*, en traitant plus en détail la question des méthodes de traitement d'images automatiques qui nous intéressent tout particulièrement dans ce manuscrit.

Dans le **Chapitre** 2, nous décrivons tout d'abord les concepts fondateurs de l'*apprentissage automatique* et de l'*apprentissage profond* dans le contexte du traitement d'images. Nous abordons par la suite les notions d'*Apprentissage par transfert*, d'*Apprentissage auto-supervisé* et d'*Apprentissage de métriques*, qui sont des concepts importants pour la méthode que nous proposons.

Le **Chapitre** 3 décrit les protocoles d'apprentissage et d'évaluation que nous proposons pour notre tâche qui est de proposer des suggestions d'appairages de fragments. Nous discutons des deux bases de données de documents sur lesquelles nous avons mené nos expériences, leurs caractéristiques et les éventuels pré-traitements qui ont été nécessaires à leurs utilisations. Des premières expériences ayant pour but de déterminer si l'approche que nous proposons est pertinente sont exposées. Ensuite, nous appliquons la méthodologie que nous avons mise en place sur les deux bases de données, avec trois modèles basés sur *VGG16*, *ResNet50* et une architecture inspirée de *VGG* que nous avons appelée *Papy-S-Net*.

Le **Chapitre** 4 décrit la manière dont nous mettons en place des méthodes d'*Apprentissage par transfert* et d'*Apprentissage auto-supervisé* pour pouvoir proposer des suggestions d'appairages de fragments dans des contextes ou peu, voire aucune donnée annotée ne sont disponibles. Nous y entraînons et évaluons nos approches sur les deux jeux de données décrits au Chapitre 3. Nous comparons les performances avec les résultats précédents et discutons de la pertinence de nos différentes propositions en fonction des cas de figure de données qui peuvent se présenter. Nous explorons aussi la possibilité d'améliorer les performances de nos modèles en combinant à de l'augmentation de données un processus de sélection dynamique des exemples difficiles au sein des batchs d'entraînement.

Enfin, dans le **Chapitre** 5, nous appliquons les méthodes développées dans cette thèse sur un cas concret, la collection de papyrus du projet *GESHAEM*. Nous décrivons la collection et le processus mis en place pour l'acquisition numérique des

papyrus. Nous appliquons ensuite une combinaison d'*apprentissage auto-supervisé*, d'apprentissage par transfert et de sélection dans les batchs avec augmentation de données pour calculer des suggestions d'appairages des fragments de la collection *Jouguet*. Ce chapitre se conclut par une analyse quantitative et qualitative des résultats en collaboration avec Marie-Pierre Chaufray, papyrologue en charge du projet *GESHAEM*.

# Chapitre 1

# Papyrologie Numérique

## Sommaire

# Introduction

La *Papyrologie* est un domaine d'étude qui s'intéresse aux manuscrits écrits sur papyrus. Cette discipline s'intéresse principalement aux documents provenant de divers sites d'Égypte écrits en grec, latin ou démotique, mais ne s'y limite pas. La *Papyrologie* cherche à étudier le contenu des documents dans un but de compréhension des sociétés anciennes.

La *Papyrologie* s'inscrit donc plus largement dans l'étude de civilisations anciennes à travers des documents qui ont été retrouvés au cours de fouilles archéologiques comme celles menées par *Pierre Jouguet* au début du XXe siècle. L'étude de ces documents nécessitait à cette époque la présence physique des artefacts, ce qui posait des problèmes de conservation, de difficulté d'accès et de centralisation, certains corpus étant dispersés dans différentes collections à travers le monde (Gascou (2001)). Ainsi, la numérisation de ces documents est aujourd'hui capitale pour la communauté des chercheurs en *Papyrologie*, car elle résout en partie ces problèmes, tout en apportant de nouvelles possibilités rendant le travail des chercheurs plus efficace. De nombreuses collections ont aujourd'hui été numérisées ou sont en cours de numérisation. Les objectifs de ces collections numériques peuvent être de centraliser des corpus dispersés dans différentes collections physiques ou de proposer une liste la plus exhaustive possible des documents d'une certaine époque, dans une langue précise ou provenant d'une région géographique particulière.

Plus généralement, on appelle **Papyrologie Numérique** (*Digital Papyrology* en anglais) l'ensemble des ressources et méthodes numériques permettant la création, le stockage, l'indexation, le traitement et la publication d'informations concernant le domaine de la *Papyrologie*. La **Papyrologie Numérique** est l'opportunité d'utiliser des méthodes informatiques pour traiter automatiquement ou semi-automatiquement ces données.

Dans ce chapitre, nous explorerons les projets de numérisation de papyrus les plus importants, ainsi que l'impact qu'a eu cette informatisation sur le domaine de recherche qu'est la *Papyrologie*. Ce changement de paradigme a en effet significativement changé la façon dont travaillent les papyrologues, et des découvertes importantes ont pu être réalisées grâce à l'utilisation de méthodes de photographie numérique et de traitement automatique d'images.

## 1.1 De la papyrologie traditionnelle à la papyrologie numérique

Après une brève description de ce qu'est la papyrologie, ses origines et enjeux, nous nous attacherons à comprendre comment et pourquoi cette discipline s'est progressivement tournée vers une informatisation croissante de ses pratiques.

### 1.1.1 Papyrologie Traditionnelle

Selon l'*Oxford Handbook of Papyrology* (Bagnall (2011)), la papyrologie est "*[...] une discipline qui s'attache à la découverte et à l'exploitation d'artéfacts anciens portant de l'écriture, ainsi qu'au contenu textuel préservé sur ces artéfacts.*" Il est intéressant de noter que la plupart du temps, la papyrologie s'intéresse à des textes "*de tous les jours*" plutôt qu'à des écrits officiels prévus pour durer dans le temps, ces derniers étant habituellement gravés dans la pierre ou le métal. Pour des raisons environnementales et climatiques, la plupart des papyrus qui ont été retrouvés proviennent d'Égypte. Presque 80% des papyrus qui ont été retrouvés et publiés à ce jour sont écrits en grec et latin, avec une dominance de documents écrits à l'époque de l'Empire romain. Ainsi, la papyrologie était historiquement définie comme l'étude des documents grecs et latins provenant de sites d'Égypte. Aujourd'hui, la définition est plus large et inclut d'autres langues, comme le copte et l'arabe et inclut parfois même certains textes écrits sur d'autres supports que le papyrus.

Le chapitre 2 de l'*Oxford Handbook of Papyrology* donne beaucoup de détails sur l'histoire de la papyrologie. Ce n'est pas le sujet de ce manuscrit, mais on peut retenir les points clés suivants. On peut retracer l'origine probable de la discipline en 1752, lorsque les premiers papyrus furent découverts à *Herculaneum*, une ville romaine antique détruite par l'éruption du Vésuve en 79 CE. Les premières découvertes archéologiques significatives de papyrus égyptiens datent de l'expédition de Napoléon Bonaparte en Égypte entre 1798 et 1801. Par la suite, avec la modernisation de l'Égypte sous *Mohamad Ali* entre 1805 et 1848 ouvrant le pays à l'influence occidentale, est née une ferveur archéologique pour la région. Plusieurs pays comme la France, l'Angleterre, l'Irlande ou l'Italie financent des fouilles archéologiques afin de découvrir de précieuses antiquités pour fournir leurs musées. Des agents privés amassent aussi parfois d'énormes collections qu'ils vendent ensuite en Europe, principalement aux musées. La période est marquée par de nombreuses fouilles illégales, du trafic d'antiquités et de la production de faux, qu'il a parfois fallu des années pour détecter.

La papyrologie s'est toujours confrontée à d'importants problèmes de conservation de ses découvertes archéologiques. Le papyrus étant une matière organique, il est sujet au pourrissement, à la moisissure ainsi qu'à l'attaque par des insectes ou

des rongeurs. Ainsi, les documents pouvaient être dégradés avant même de se retrouver dans des conditions propices à une conservation sur des milliers d'années. Il se trouve que les papyrus qui ont été retrouvés les mieux préservés étaient conservés dans des jarres, des coffres ou des boites dans des endroits très secs. Lors des fouilles, les archéologues sortent les papyrus de ces conditions qui ont permis leur longue conservation et doivent donc veiller à les stocker de manière à éviter une plus grande décomposition.

On identifie aujourd'hui plusieurs techniques, adaptées à différents cas de figure, permettant une bonne conservation dans le temps des papyrus. Après avoir nettoyé les fragments de papyrus des restes de terre ou de sable, une étape d'identification de l'encre est nécessaire (Bagnall (2011), chapitre 4). Les produits chimiques qui seront utilisés ainsi que les conditions lumineuses de conservation dépendront en partie du type d'encre présent. Ensuite, les papyrus sont légèrement humidifiés avec de l'eau ou une solution de cellulose en fonction différents paramètres. Cette étape sert à rendre le matériau plus facile à manipuler sans le casser. S'il présentait des pliures, il est déplié puis placé entre deux feuilles de papier buvard pour absorber l'humidité. Une fois de nouveau sec, il subit une dernière phase de nettoyage, et si l'encre se trouve être "floconneuse" ("*flaky*" en anglais), elle est fixée à l'aide de *Méthylcellulose* ou de pâte d'amidon de blé.

C'est ensuite que les éventuels travaux de reconstruction des papyrus peuvent avoir lieu. En utilisant des lignes de texte qui continuent sur deux fragments (au recto ainsi qu'au verso), la couleur des fragments, le style d'écriture, les trous que des vers ont pu laisser qui peuvent se répéter dans le cas ou le papyrus était sous la forme d'un rouleau, les motifs des fibres du papyrus ou bien sûr en lisant le contenu du texte.

Enfin, les papyrus sont archivés. Ils peuvent être placés entre deux feuilles de papier buvard non acide, ou montés dans un cadre entre deux vitres anti-UV s'ils sont souvent utilisés ou qu'il est prévu de les exposer. Il est recommandé de conserver les papyrus dans un environnement dans lequel la température et l'humidité sont les plus stables possibles. En plus de toutes ces précautions, il est quand même important de limiter les interactions avec le matériel. Cependant, il va de soi que les chercheurs en papyrologie ont besoin d'un accès aux papyrus afin de pouvoir travailler. C'est une des raisons pour laquelle de grands projets d'acquisition photographique des papyrus ont été menés à travers le monde dès que la technologie l'a permis. Dès l'arrivée d'internet et avec l'essor de la photographie numérique, l'idée de créer des collections numériques a émergé dans la communauté papyrologique, ce qui a donné naissance à la papyrologie numérique. En plus de pouvoir visualiser les papyrus sans les manipuler, il est possible de rendre accessibles tous ces artéfacts aux chercheurs du monde entier, ainsi qu'au public. Nous verrons dans la section suivante comment la papyrologie numérique a fait évoluer la pratique des chercheurs en papyrologie au cours de ces dernières décennies.

### 1.1.2 Papyrologie Numérique

La numérisation des collections de papyrus a permis de faciliter différents aspects du travail des papyrologues. L'indexation des documents a pu se faire de manière informatique, permettant des recherches automatiques par mots-clés, par thèmes, par auteurs, etc. La prise de photographies infrarouge avec différentes longueurs d'onde ainsi que des techniques de traitement d'image pour améliorer l'image peuvent offrir de meilleurs supports de lecture, rendant le texte plus lisible, et même rendant un texte illisible, lisible (cf. Fig. 1.1). Ces dernières années, des travaux de recherche en informatique ont abouti au développement de méthodes permettant d'identifier les auteurs des documents ou d'identifier si deux documents ont été écrits par le même auteur (Nasir et Siddiqi (2021), Cilia et al. (2021), Sudarma (2015)). Par exemple, les manuscrits de la mer morte sont encore aujourd'hui l'objet de recherches visant notamment à identifier leurs auteurs. En 2021, *Popović et al.* ont pu mettre en évidence qu'un des rouleaux les plus importants de la collection, le *Grand Rouleau d'Isaïe* avait en fait probablement été écrit par deux scribes, contrairement à ce que concluaient de précédentes analyses expertes (Popović et al. (2021)). Cette découverte repose sur l'utilisation de méthodes de traitement d'image et d'apprentissage automatique à partir des acquisitions numériques de la collection. De même, l'évolution de la reconnaissance de caractères automatique dans les documents anciens permet d'imaginer des outils de transcription automatique dans un futur proche (Swindall et al. (2021)).

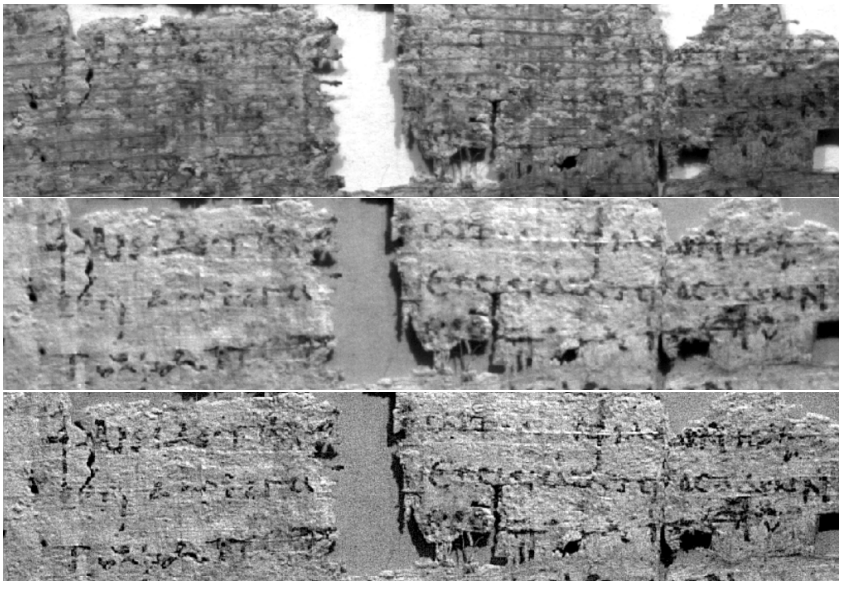


FIGURE 1.1 – Le texte écrit sur ce papyrus était quasiment invisible (première image). Grâce à une combinaison de photographie infrarouge et de légers traitements d'image numériques, Oates et al. (1999) ont pu rendre le texte lisible (deux propositions sur la deuxième et la troisième image)

"L'informatisation" de la *Papyrologie* a aussi fait naître des pratiques de travail collaboratif (*Crowdsourcing*) entre experts et non-experts du monde entier. En particulier, on peut citer le projet *Ancient Lives*[1] dans lequel est utilisée la plateforme de *Crowdsourcing Zooniverse*[2] pour transcrire des manuscrits grecs de la collection des *Papyrus d'Oxyrhynque*. De même, on peut contribuer à identifier les scribes des papyrus de la *Guéniza du Caire* grâce au projet *Scribes of the Cairo Geniza* sur *Zooniverse*[3].

## 1.2 Principales collections existantes

Nous présentons ici les projets de *Papyrologie Numérique* qui nous semblent les plus importants en termes de quantité de données et qui représentent bien la diversité des collections existantes. Pour une liste plus complète, voir Annexe A. Certaines collections que nous présentons ci-dessous sont accessibles de manière centralisée via le site papyri.info. C'est une ressource précieuse pour les chercheurs en papyrologie, il propose le *Papyrological Navigator* permettant de rechercher des documents au sein des collections et le *Papyrological Editor* permettant un travail collectif d'annotation, de traductions, d'édition de méta données, etc.

FIGURE 1.2 – Exemple de fragments de papyrus reconstruits des papyrus de la mer morte. Source : Wikimedia commons. Cette image est sous licence *Creative Commons Attribution 2.0 Generic*

1. https://www.ancientlives.org/
2. https://www.zooniverse.org/
3. https://www.zooniverse.org/projects/judaicadh/scribes-of-the-cairo-geniza

Parmi les projets de numérisation les plus connus, on retrouve bien sûr la collection des **papyrus de la mer morte**, dont on peut voir un exemple sur la Figure. 1.2. Les manuscrits de la mer morte sont un corpus de textes écrits principalement en hébreu sur du parchemin et du papyrus. Découverts entre 1947 et 1956 en Cisjordanie, ces 970 manuscrits écrits entre -300 et 100 contiennent notamment de nombreux livres de l'ancien testament chrétien. Ces textes présentent un grand intérêt pour la compréhension de l'histoire de la Bible.

Ils ont été retrouvés sous la forme d'environ 100.000 fragments, dont le travail de reconstruction a été prétexte au développement de nombreuses méthodes pour la reconstruction de fragments anciens. Plusieurs campagnes d'acquisition d'images ont été menées à différentes époques par différentes organisations. D'abord avec de la photographie argentique puis numérique.

Les premières photographies datent de 1948, mais c'est entre 1952 et 1967 qu'une réelle entreprise d'acquisition à grande échelle a été mise en place par le musée d'archéologie de Palestine. Durant cette période, 1750 photographies infrarouges ont été prises, et un protocole de tri et d'assemblage des fragments a été mis en place. Aujourd'hui, ces photographies sont les meilleurs supports de lecture pour les chercheurs, car les rouleaux originaux ont depuis été dégradés par les conditions de conservation de l'époque.

Entre 1993 et 2002, la *NASA* a entrepris de produire des photographies infrarouges numériques des fragments de la mer morte. Puis entre 2011 et 2016, un partenariat entre Google et le musée de Jérusalem d'Israël a été lancé afin de continuer l'acquisition numérique haute définition ainsi que d'élaborer une plateforme web concentrant les acquisitions et les informations sur les documents, tout en la rendant accessible au public [4].

Plus modeste, la *Banque des images des papyrus de l'Aphrodité byzantine* (le **BIPAb**) a pour objectif de regrouper en un seul endroit toutes les images de papyrus grecs et coptes du village d'Aphrodité (les *archives de Dioscore d'Aphrodité*), dont les originaux sont dispersés dans plusieurs collections. Ces 650 documents datent du VIe siècle EC et constituent le plus important corpus de papyrus de l'époque byzantine. Cette collection fait encore aujourd'hui l'objet de recherches afin d'identifier les auteurs de certains documents [5].

On peut aussi citer le projet de la *Berlin Papyrus database* (**BerlPap**) qui a pour objectif de numériser et d'indexer les papyrus écrits en grec et latin de la collection de papyrus du *Staatliche Museen zu Berlin*. La collection contient des dizaines de milliers de papyrus écrits, 7000 ostraca, plus de 1000 parchemins et de nombreux autres documents écrits sur d'autres supports tels que le papier, des textiles, du bois,

4. http://dss.collections.imj.org.il/
5. http://bipab.aphrodito.info/

des tablettes de cire, du cuir ou des tablettes de plomb. Au moment de la rédaction de ces lignes, la base de données en ligne donne accès à 4974 papyrus numérisés, le travail d'acquisition numérique est toujours en cours [6].

Toujours en Allemagne, la collection de papyrus de l'Université d'Heidelberg est composée de 10.500 papyrus, parchemins, supports papiers et ostraca égyptiens. C'est la deuxième plus grande collection d'Allemagne. Les documents sont écrits en grec, copte et latin. L'intégralité de la collection a été numérisée entre 1999 et 2002, et des transcriptions des textes sont disponibles [7].

Une autre collection numérique importante, *The Arabic Papyrology Database* comporte 12.789 documents allant du VIe siècle EC au XVIIe siècle EC. Ils sont écrits sur différents supports comme le papyrus, le parchemin ou encore le papier. C'est un projet international qui rassemble des papyrus provenant de collections localisées dans 30 pays différents [8].

Différents projets visent aussi à construire des Paléographies numériques, comme le projet *DPDP* [9] de l'Université d'Heidelberg dont la collection est composée de 10.500 papyrus, parchemins, supports papiers et ostraca égyptiens. C'est la deuxième plus grande collection d'Allemagne. Les documents sont écrits en grec, copte et latin. L'intégralité de la collection a été numérisée entre 1999 et 2002, et des transcriptions des textes sont disponibles [10]. Dans le même esprit, mais sur des supports différents, on peut citer le projet *DigiPal* [11], destiné à l'écriture manuscrite médiévale produite en Angleterre entre les années 1000 et 1100. Son interface permet de facilement comparer visuellement des échantillons d'écriture manuscrite entre eux.

Les avancées dans le domaine de la *Papyrologie Numérique* ont ainsi donné naissance à de nombreux projets interdisciplinaires entre papyrologues et informaticiens. La communauté des chercheurs en papyrologie a pu bénéficier d'outils rendant son travail plus efficace en permettant un meilleur accès aux données de la communauté internationale, des opportunités de travail collaboratif entre experts et non-experts, des techniques de traitement d'image améliorant la lisibilité du texte, etc.

Il faut noter que les collections décrites ci-dessus ont pour la plupart été conçues indépendamment et au fil des années. Leurs projets de numérisation ont été lancés à des époques différentes, avec du matériel différent et des protocoles d'acquisition hétérogènes. Malgré une volonté très ancienne de standardisation des pratiques de publication en papyrologie traditionnelle, pouvant remonter jusqu'en 1936 avec les

6. https://berlpap.smb.museum/
7. http://www.rzuser.uni-heidelberg.de/~gv0/
8. https://www.apd.gwi.uni-muenchen.de/apd/project.jsp
9. www.uni-heidelberg.de/fakultaeten/philosophie/zaw/aegy/forschung/dpdp.html
10. http://www.rzuser.uni-heidelberg.de/~gv0/
11. http://www.digipal.eu/

principes proposés par *Calderini* (Reggiani (2017)), et malgré le fort intérêt porté par les papyrologues à l'informatisation de leurs pratiques, la transition au numérique n'a pas donné lieu à une uniformisation globale de toutes les collections. Un grand effort dans ce sens est en cours avec la plateforme papyri.info qui regroupe déjà 5 grandes bases de données (APIS, DDbDP, HGV, BP et Trismegistos) et bientôt une sixième (APD).

En plus de ces collections de tailles significatives, il faut ajouter la collection de papyrus de l'Université du Michigan [12], dont nous nous sommes servis pour nos expériences et que nous présentons dans la section 3.3.1 Il existe aussi beaucoup de collections de tailles plus modestes qui ne sont pas numérisées ou sont en cours de numérisation, comme celle du projet *GESHAEM*.

### Le Projet GESHAEM

FIGURE 1.3 – Exemple de cartonnage particulièrement bien conservé, provenant du projet *GESHAEM*.

12. https://www.lib.umich.edu/locations-and-hours/papyrology-collection

Le projet *GESHAEM*[13], dans lequel s'intègrent ces travaux de thèse à travers le financement de l'*ERC*[14] obtenu par l'équipe de chercheurs du laboratoire Ausonius[15], vise à étudier des archives administratives qui ont été recyclées pour fabriquer des cartonnages de momies égyptiennes datant de l'époque hellénistique. Les cartonnages sont des ornements funéraires conçus à base de feuilles de papyrus collées pour créer une forme et peints (cf. Fig. 1.3). À l'époque, le papyrus était une ressource coûteuse. Il était alors courant de réutiliser des papyrus qui ne pouvaient plus servir de support d'écriture, car on avait déjà écrit dessus et dont le contenu n'avait plus d'intérêt, pour confectionner ces cartonnages.

Ces documents font partie de la collection *Jouguet* de la Sorbonne, qui est à ce jour méconnue et peu étudiée. Ils sont le résultat de fouilles réalisées par *Pierre Jouguet* dans le Fayoum en 1901 et 1902. Ces archives contiennent beaucoup de documents fiscaux. On y retrouve par exemple des listes de contribuables ayant ou devant payer des impôts en argent ou en blé, lettres administratives, contrats, etc. Le projet a principalement pour objectif d'évaluer le rôle des Égyptiens dans l'administration de cette période. En effet, jusqu'à présent les historiens ont surtout parlé du rôle des Grecs dans cette administration en considérant que l'Égypte était devenue un royaume grec. Mais en fait, les Égyptiens se sont pleinement investis dans la nouvelle administration, aux échelons inférieurs, au niveau du village : ils avaient un savoir-faire dans le domaine agricole, la spécificité de l'Égypte étant que l'agriculture se fait au rythme des crues du Nil, qu'ils ont continué à utiliser pour aider la nouvelle administration à contrôler les récoltes et le montant des impôts qui étaient prélevés sur ces récoltes.

En particulier, le projet *GESHAEM* compte étudier trois corpus de textes. Des **contrats de cautionnements** rédigés en égyptien, par des scribes égyptiens, c'est-à-dire des villageois se portant garant pour d'autres villageois qui s'engagent à prélever des impôts sur des produits comme de la bière, la blanchisserie, etc. Des **registres agricoles** dans lesquels sont notés les noms des paysans qui cultivent les terres, le type de culture qu'ils doivent faire, les surfaces à ensemencer et les impôts qu'ils doivent payer proportionnellement. Et, enfin, des **registres de taxes** en argent. Tous ces textes sont rédigés en démotique et montrent donc l'implication des Égyptiens dans l'établissement de ces registres essentiels à l'administration fiscale.

La base de données du projet contient 381 fragments écrits en démotique au recto et souvent en grec au verso. Les fragments sont photographiés en couleur ainsi qu'en infrarouge. Nous donnons plus d'informations sur le protocole d'acquisition dans la section 5.1.2.

13. https://geshaem.huma-num.fr/
14. European Research Council
15. https://ausonius.u-bordeaux-montaigne.fr/

## 1.3 Papyrologie et traitement d'image

Les collections numérisées sont des sources de données très intéressantes pour la communauté informatique travaillant sur l'analyse de documents anciens. Elles présentent cependant des difficultés pour être exploitées informatiquement. Bien que des initiatives existent, il n'y a pas encore de standard utilisé universellement pour le processus d'acquisition, ni pour la façon dont sont stockées les images et leurs méta-données et il est rare d'avoir accès à une Application Programming Interface (API) permettant de recueillir les données des collections facilement. Tout cela rend souvent la conception de bases de données de travail assez difficile. Cela vient du fait que quasiment toutes ces bases de données sont pensées pour être utilisées par des papyrologues, et même par des papyrologues spécialisés dans un sous-domaine particulier.

Les auteurs d'une récente étude (Philips et Tabrizi (2020)) identifient cinq phases pour le traitement des documents anciens (cf. Fig. 1.4).

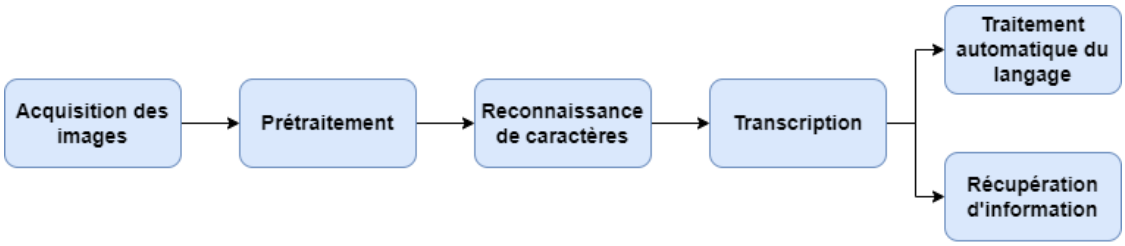



FIGURE 1.4 – Les cinq phases pour le traitement des documents anciens (Philips et Tabrizi (2020)).

Nous parlons donc dans la suite des différents prétraitements utilisés dans le domaine de la papyrologie numérique, ainsi que des méthodes de reconnaissance de caractères qui pourront rendre possible la transcription des papyrus/ Cependant, comme le soulignent Philips et Tabrizi (2020) :

*"[...] peu, voire aucune études n'ont été publiées sur des algorithmes de transcription automatique, des outils logiciels ou des jeux de données de référence pour ces documents anciens qui précèdent l'époque médiévale. Très fragmentaires et détériorés, les papyrus seront probablement encore un défi pour la transcription automatique dans un futur proche."*

Il existe encore aujourd'hui relativement peu de travaux informatiques spécifiques aux papyrus. On trouve beaucoup de méthodes pour la transcription, identification d'auteurs, datation, restauration de documents, etc. appliquées à des documents anciens en général, mais encore peu sur le support précis que sont les papyrus. Nombre de ces travaux sur d'autres supports d'écriture pourraient certainement

être appliqués aux papyrus, c'est pourquoi dans les différents sous-problèmes exposés ci-dessous, nous décrivons aussi des méthodes ayant été appliquées sur d'autres supports que les papyrus. De plus en plus de travaux exploitent la puissance du *Deep Learning*. C'est un domaine regroupant un ensemble de méthodes d'apprentissage automatique et exploitant principalement les réseaux de neurones profonds. Une étude récente de Lombardi et Marinai (2020) dresse à ce sujet un état des lieux des approches *Deep Learning* utilisées dans les différentes étapes de l'analyse informatique des documents anciens.

### 1.3.1 Prétraitement

#### Restauration numérique des documents

Certains travaux se concentrent sur la "restauration" numérique des papyrus dans le but principal de rendre le texte plus visible. L'article Sparavigna (2009) présente une méthode pour faire ressortir les lettres des papyrus en utilisant de la détection de contours et des filtres de Fourier. Atanasiu et Marthot-Santaniello (2021) font le constat qu'il existe peu de méthodes purement informatiques se basant uniquement sur des photographies. Les méthodes donnant les meilleurs résultats combinent de la photographie multispectrale et des post-traitements informatiques. Ils proposent cependant de nouvelles méthodes basées sur des illusions visuelles et du traitement des couleurs qui améliorent la lisibilité du texte pour les humains. Sudarma (2015) proposent une approche permettant de séparer les caractères du fond sur des papyrus Balinais anciens en les projetant dans l'espace de couleur *CIELab*.

Pour préparer les images à d'autres traitements, il est parfois nécessaire d'enlever un maximum de bruit dû au processus d'acquisition photographique. Sur ces types d'images, utiliser de simples filtres tels qu'un filtre médian ou moyen n'est pas suffisant. En effet, ils font souvent disparaître certains détails importants. Ces dernières années, l'état de l'art a convergé vers l'utilisation de méthodes d'apprentissages profondes, Nguyen et al. (2021b) utilisent par exemple une architecture encodeur-décodeur associée à un mécanisme d'attention pour réduire le bruit présent dans des photographies d'anciennes stèles.

#### Analyse de la disposition et segmentation

L'analyse de la disposition consiste à identifier et à étiqueter l'organisation physique des documents. On y retrouve principalement la détection de tableaux (cf. Fig. 1.5), la segmentation du texte ou la détection des sous-lignes du texte et la segmentation des paragraphes (cf. Fig. 1.5). À part la segmentation et la détection des sous-lignes du texte, ces étapes sont rarement utiles dans le cadre de l'analyse de papyrus, car ces derniers ont la plupart du temps une structure très simple, à savoir des lignes de texte écrites sur toute la largeur du document. Nous n'avons trouvé

aucun travail appliqué directement aux papyrus en ce qui concerne la segmentation/détection de lignes. Il existe par contre de nombreuses propositions appliquées aux documents manuscrits, dont une que nous avons pu appliquer directement sur des papyrus, comme celle proposée par Grüning et al. (2019).

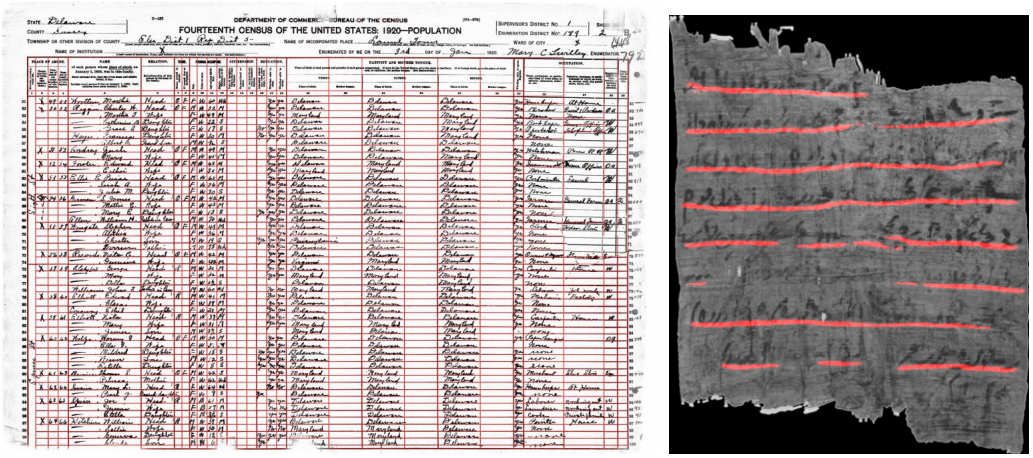


FIGURE 1.5 – À gauche, exemple de détection de la structure d'un tableau dans un document historique. (Source : Lehenmeier et al. (2020)). À droite, exemple de détection de sous-ligne sur un papyrus.

Dans un article de 2019, Grüning et al. (2019) proposent une méthode en deux étapes d'extraction de sous-lignes sur des documents manuscrits anciens. La première étape effectue de la segmentation de pixels, c'est-à-dire une classification binaire des pixels en deux classes : sous-lignes ou non. On peut voir un exemple de sous-ligne sous la forme des lignes rouges présentes sur la Figure. 1.5 à droite. La seconde étape utilise des techniques de traitement d'image plus traditionnelles pour extraire de cette segmentation des sous-lignes "géométriques" précises. Nous expliquerons comment nous avons utilisé cette méthode pour détecter les lignes dans nos papyrus dans la section 3.3.3.

### Reconstruction automatique de documents fragmentaires

La reconstruction des documents fragmentaires peut être considérée comme une étape de prétraitement dans le sens qu'il est nécessaire de l'effectuer avant l'analyse de ces documents par des experts. Cette étape, appliqué aux papyrus se trouve être encore moins traité que les autres dans la littérature. En effet, à l'exception de travaux de recherche au sein de notre équipe (Pirrone et al., 2021, 2019) et (Ostertag et Beurton-Aimar, 2021, 2020)) dont un portant sur la reconstruction d'ostraca, on trouve peu de travaux sur ce domaine précis dans la littérature. On peut cependant citer un article de juillet 2021 (Abitbol et al. (2021)) utilisant une approche similaire à la nôtre et faisant le même constat concernant la rareté des publications à ce sujet.

Il existe cependant quelques autres travaux en cours, comme, le projet *Crossing Boundaries*[16] de l'Université de Bâle, qui étudie des textes de la communauté égyptienne ancienne *Deir el-Medina* et a pour but d'utiliser des méthodes de *Machine Learning* pour la classification et la reconstruction de leurs fragments de papyrus. C'est une partie du projet thèse de *Stephan Unter*[17], en plus de développer un logiciel pour manipuler virtuellement les fragments, la *Virtual Light Table*[18].

Il est intéressant de remarquer que d'autres domaines de recherche s'intéressent à ces documents, par exemple des chercheurs israéliens ont exploité l'ADN des parchemins de la mer morte (qui sont produits à partir de peau d'animaux) en identifiant des espèces d'animaux et même des variations génomiques individuelles au sein de différents fragments. Il a ainsi été possible d'associer de nouveaux fragments et de corriger de précédentes erreurs d'assemblage (Anava et al. (2020)).

### 1.3.2 Reconnaissance et classification d'écriture

Le même constat d'un manque important de travaux dans la littérature concernant l'identification et la classification de caractères anciens provenant **spécifiquement de papyrus** est fait par Haliassos et al. (2020). Pourtant, si on se base sur le nombre de contributions très récentes, notamment lors de la dernière édition de *ICDAR*[19] en 2021, ce domaine semble très actif. Wick et Reul (2021) travaillent par exemple sur la reconnaissance de caractères sur des documents imprimés anciens. Leur approche consiste à optimiser un ensemble de modèles en même temps, à la fois indépendamment et en fusionnant leurs sorties. Grâce à cette méthode, il est possible d'obtenir un modèle efficace sur un livre donné alors que seulement quelques lignes de vérité terrain étaient disponibles. Scius-Bertrand et al. (2021) proposent une méthode pour localiser les caractères sur des stèles vietnamiennes anciennes sans annotation humaine, utilisant un processus d'autocalibration et des méthodes de détections d'objets entraînées sur des caractères imprimés. Dans les domaines dans lesquels il existe peu de données annotées, Kišš et al. (2021) proposent une stratégie de *domain adaptation* en *self-training* en utilisant des données massivement annotées de domaines connexes. Ils se placent dans le scénario dans lequel une petite partie du domaine cible est annoté et une grande partie du domaine connexe est annoté.

### 1.3.3 Identification d'auteurs

Le domaine de l'identification d'auteurs (*Scribe identification*), consiste comme son nom l'indique à identifier les auteurs de documents, souvent en évaluant si deux

16. http://web.philo.ulg.ac.be/x-bound/
17. http://web.philo.ulg.ac.be/x-bound/portfolio-item/stephan-unter/
18. http://web.philo.ulg.ac.be/x-bound/virtual-light-table/
19. https://icdar2021.org/

documents ou fragments de documents ont été écrits par le même auteur. Cela permet aussi parfois d'implicitement estimer l'origine géographique et la date de rédaction du document He et al. (2016).

Comme dans beaucoup de domaines, l'identification d'auteurs se tourne de plus en plus vers des approches orientées *Deep Learning*. Nasir et al. (2021) proposent une approche basée sur l'extraction de patchs spécifiques, dont le choix dépend de points d'intérêts issus de l'extracteur de caractéristiques *FAST* (Rosten et Drummond (2006)). Ces patchs sont choisis en fonction de leurs probabilités de contenir de l'information dite *"writer specific"*, c'est-à-dire qui est susceptible de caractériser l'écriture d'un scribe. Les auteurs entraînent trois modèles d'apprentissage profond (*VGG16*, *Inception v3* et *Resnet50/34*, cf. Section. 2.2.2) avec deux étapes de ré-entraînement. Les modèles sont d'abord pré-entraînés sur une des bases de données de reconnaissance d'images de référence, *ImageNet*[20], puis ré-entraînés avec des données contenant de l'écriture manuscrite moderne, puis avec des données contenant de l'écriture manuscrite sur papyrus.

Pareillement, He et Schomaker (2020) proposent une architecture de réseaux de neurones profonds, *FragNet* pour caractériser les auteurs à partir de petits exemples de leurs écritures. Les auteurs entraînent leurs modèles à la fois sur des images de mots entiers, et sur des sous-images de ces mots afin de les forcer à explorer toute l'information du style d'écriture contenue dans les exemples. Cela se traduit par deux "chemins" qu'empruntent les deux types d'entrées possibles. Le premier est appelé *"feature pyramid"*. Il prend en entrée les images entières de mots et est composé d'enchaînements de couches de convolutions, de *pooling* et de *batch normalization*. Le deuxième est appelé *fragment pathway*. Il prend en entrée les sous-images extraites de l'image du premier chemin et a la particularité de ré-extraire de l'autre chemin une sous-image, mais dans son espace des caractéristiques et de l'utiliser à différentes étapes de son déroulement (cf. Fig. 1.6).

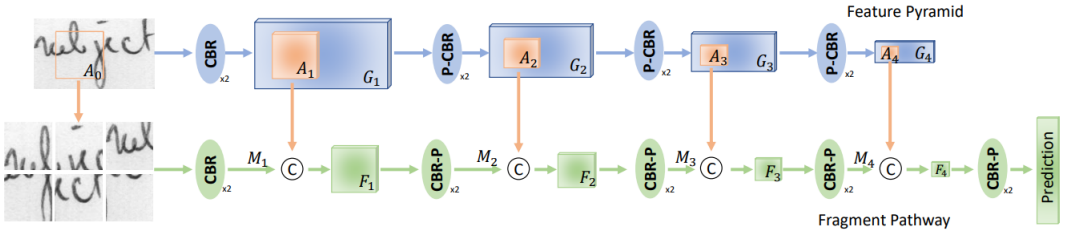



FIGURE 1.6 – Architecture de *FragNet*. Source : He et Schomaker (2020)

Déjà très actif, ce domaine pourrait prendre encore de l'ampleur dans les prochaines années avec de nouveaux jeux de données comme celui de la compétition *HisFrag* Seuret et al. (September, 2020) qui est une base synthétique de fragments

20. https://www.image-net.org/

de documents anciens (plus de détails dans la section 3.3.2), ou avec *PapyRow*, Cilia et al. (2021) qui propose un jeu de données de papyrus grecs pour l'identification des auteurs.

## Conclusion

La transition d'une papyrologie traditionnelle à la papyrologie numérique que l'on connaît aujourd'hui s'est ainsi effectuée au travers d'évolutions successives des méthodes de conservation, de numérisation, de traitement et d'analyse des images. Que ce soit dans des buts de restauration ou d'extraction d'information, l'informatisation croissante de la pratique a aujourd'hui démontré son intérêt pratique pour les papyrologues ainsi que son intérêt en tant qu'objet de recherche pour les informaticiens.

Nous nous sommes concentrés ici sur la papyrologie, car c'est le domaine applicatif final dans lequel s'inscrivent ces travaux de thèse. Cependant, nombre des informations données dans ce chapitre s'appliquent aussi aux documents historiques et même aux documents en général. Ainsi, dans la suite de ce manuscrit, nous exposerons de manière plus générale nos propositions appliquées aux documents, et non seulement aux papyrus. Nous retournerons à une application plus orientée pour les papyrologues dans le chapitre 5, qui présente entre autres une application de nos travaux aux papyrus du projet *GESHAEM* en collaboration avec des papyrologues du projet.

# Chapitre 2

# Apprentissage Automatique et Apprentissage Profond

**Sommaire**

# Introduction

Beaucoup de domaines de l'informatique ont subi ces dernières décennies un important changement de paradigme avec le succès grandissant des algorithmes d'apprentissage automatique, puis d'apprentissage profond. Les performances de ces derniers ont rapidement dépassé celles des algorithmes spécifiques confectionnés dans le but de résoudre un problème précis.

Un des domaines dans lequel ce changement de paradigme s'est peut-être le plus fait ressentir est celui du traitement d'images. Avec l'arrivée des réseaux de neurones convolutifs, des immenses bases de données annotées, des puissances de calcul et des capacités en mémoire suffisantes, l'apprentissage profond a démontré sa puissance en triomphant progressivement des différents benchmarks de classification d'images [1], de reconnaissance d'objets [2], de segmentation sémantique [3] etc.

Le constat de tous ces succès a fait que nous avons choisi dans cette thèse de nous tourner vers ces approches d'apprentissage profond pour proposer des suggestions d'appairages de fragments de documents anciens. Nous commençons donc dans ce chapitre par décrire les concepts fondateurs de l'apprentissage automatique "traditionnel". Nous verrons ensuite plus en détail ce qu'est l'apprentissage profond et ses différentes implémentations dans le cadre du traitement d'images et l'apprentissage par transfert ou *auto-supervisé*. Enfin, nous détaillerons ce qu'est l'apprentissage de métriques, car c'est de ce domaine que s'inspirent nos contributions à travers les réseaux siamois.

1. https://paperswithcode.com/task/image-classification
2. https://paperswithcode.com/task/object-detection
3. https://paperswithcode.com/task/semantic-segmentation

## 2.1 Apprentissage automatique

L'apprentissage automatique, ou *Machine Learning* en anglais, vise à apprendre à résoudre des problèmes complexes automatiquement, directement à partir des données. On distingue trois grandes catégories d'algorithmes d'apprentissage automatique : l'apprentissage supervisé, l'apprentissage non-supervisé et l'apprentissage par renforcement. Nous n'abordons pas ici la question de l'apprentissage par renforcement, car nous ne nous en servons pas et que c'est un paradigme d'apprentissage qui est encore très peu utilisé dans le domaine du traitement d'images.

### 2.1.1 Apprentissage Supervisé

Un algorithme d'apprentissage supervisé prend des données en entrée, on lui spécifie la sortie désirée et il apprend une fonction associant correctement une entrée avec une sortie. Par exemple, si on cherche à classifier des images en deux catégories : *oiseau* et *pas oiseau*, une entrée peut être une image d'oiseau, la sortie désirée est la classe *oiseau*. Idéalement, au fur et à mesure de l'apprentissage, l'algorithme aura été exposé à beaucoup d'exemples des deux classes et sera capable de les classifier correctement. L'apprentissage supervisé nécessite une grande quantité de **données annotées**, qui constitue la vérité terrain, pour atteindre des performances acceptables. En effet, on ne souhaite pas seulement que le modèle appris soit performant sur les données d'entraînement, mais aussi sur de nouvelles données qu'il n'a encore jamais vues. Pour cela, il est important de multiplier les exemples afin que le modèle *généralise* les concepts qu'il apprend, et non qu'il apprenne spécifiquement les données d'entraînement. On peut distinguer deux grands types de prédictions que peuvent produire des algorithmes d'apprentissage : la classification qui correspond à prédiction de probabilité sur des classes, et la régression qui correspond à la prédiction d'une fonction continue.

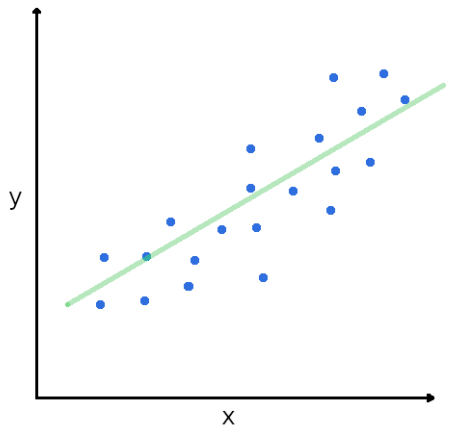



FIGURE 2.1 – Illustration d'une régression linéaire

La méthode de régression la plus élémentaire est la *Régression Linéaire*, dans laquelle on cherche à déterminer une relation entre des variables indépendantes en trouvant les paramètres d'une fonction affine de type $y = ax + b$ qui "colle" aux données (cf. Fig. 2.1). Avec une nouvelle donnée $x$, on peut prédire la valeur de $y$ en résolvant la fonction affine pour $x$.

Bien sûr, ce type de modèle ne capture pas du tout les potentielles caractéristiques non linéaires des données sur lesquelles on travaille et ne fonctionnera pas du tout dans ces cas. On peut alors utiliser des régressions plus complexes avec plus de paramètres et étant capable de capturer la non-linéarité des données, comme une *régression polynomiale*.

Pour effectuer des prédictions sur des classes plutôt que sur des valeurs continues, on se tourne vers d'autres méthodes comme les *Support Vector Machine (SVM)* (Boser et al. (1992)) par exemple. Chaque donnée est ici représentée par un vecteur de caractéristiques en *n dimensions*. On projette les données dans un espace à *n dimensions*, chaque axe correspondant à une caractéristique (cf. Fig. 2.2 à gauche). On cherche enfin l'*hyperplan* qui sépare le mieux les données en deux groupes distincts avec la marge la plus grande possible des deux côtés, sur la Figure. 2.2 (à gauche), cet *hyperplan* est représenté par la ligne noire.

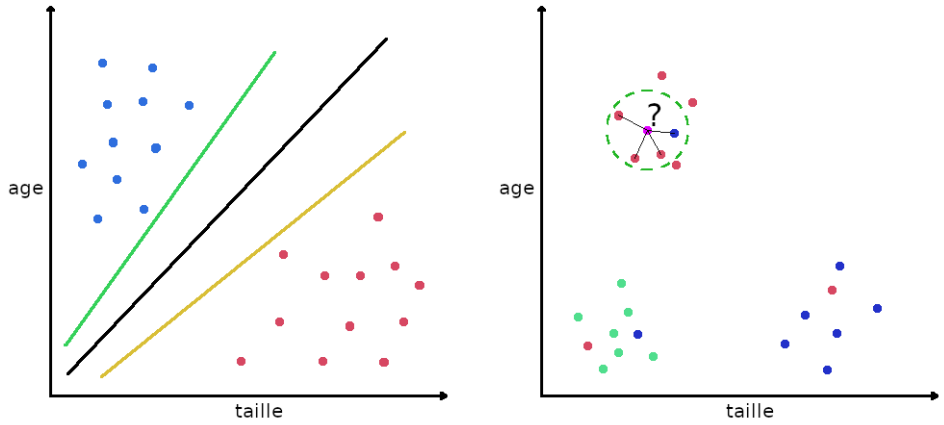



FIGURE 2.2 – Illustration d'un SVM (à gauche) et de l'algorithme des kNN (à droite).

Les données ne sont pas toujours linéairement séparables, il est possible dans ces cas de transformer l'espace de représentation des données en un espace de plus grande dimension, probablement linéairement séparable, à l'aide d'une fonction de noyau (*kernel function*).

L'algorithme des *k-Nearest Neighbors (kNN)* (Fix et Hodges (1951)), pour "k plus proches voisins" permet aussi la classification des données, mais de manière

différente. Comme pour les SVM, on représente les données dans un espace à $n$ *dimensions*, chaque dimension correspondant à une caractéristique des données. Lorsqu'une nouvelle donnée arrive (le point violet dans la Figure. 2.2 à droite), on calcule sa distance à ses $k$ plus proches voisins et on attribue au nouveau point la classe majoritaire dans ses voisins les plus proches.

Les algorithmes que nous venons de voir nécessitent qu'une quantité importante de **données annotées** soient disponibles pour fonctionner. Il existe des approches dans lesquelles cette nécessité n'existe pas, on appelle cela l'apprentissage non supervisé.

### 2.1.2 Apprentissage non supervisé

Un algorithme d'apprentissage non supervisé essaie de trouver une structure sous-jacente dans des données **non annotées** en identifiant des caractéristiques communes. Le *clustering* est un ensemble de méthodes d'apprentissage non supervisé dans laquelle des *clusters* de données sont construits automatiquement, on peut ensuite classifier une nouvelle donnée en cherchant de quel cluster elle est la plus proche.

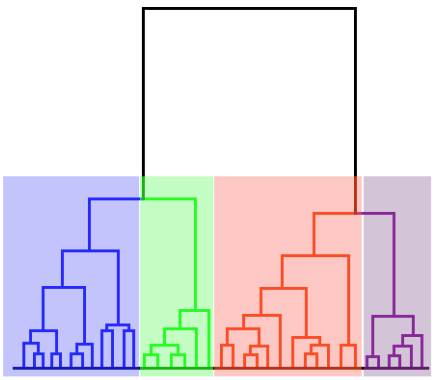


FIGURE 2.3 – Illustration d'une classification ascendante hiérarchique

Parmi les algorithmes de clustering, on peut retrouver par exemple la classification ascendante hiérarchique (Johnson (1967)), illustrée sur la Figure. 2.3 et l'algorithme des *k-means*(MacQueen et al. (1967)). Ce dernier construit itérativement $k$ clusters en calculant des distances entre les éléments. L'algorithme sélectionne initialement $k$ points au hasard et les considère chacun comme étant le centre d'un cluster. Chaque autre point est ensuite classifié comme appartenant au cluster du centre de cluster le plus proche de lui. Une fois tous les points attribués à un cluster,

on met à jour les centres en calculant la moyenne de chaque cluster, et on répète les opérations jusqu'à convergence (cf. Fig. 2.4).

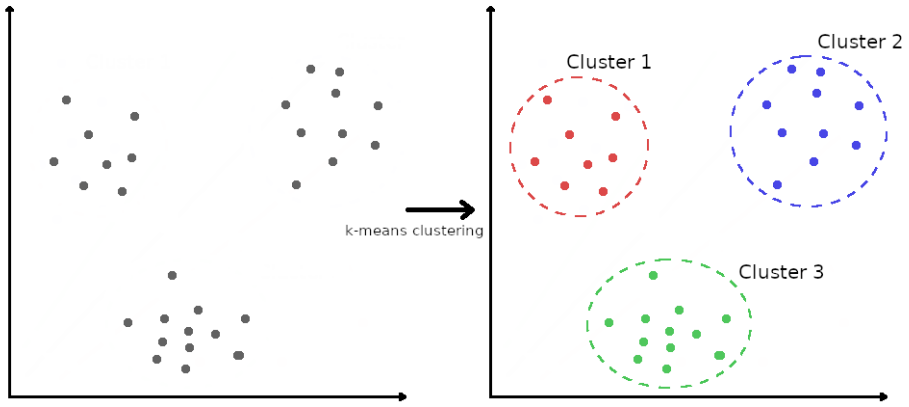



FIGURE 2.4 – Illustration d'un clustering par l'algorithme des k-means

Dans le cas d'une approche *Machine Learning* "traditionnelle", la création des vecteurs de caractéristiques est faite "à la main", c'est-à-dire qu'on doit explicitement déterminer quelles caractéristiques des données seront utilisées pour l'entraînement et la prédiction, on définit des *variables explicatives*. Ce n'est pas un problème lorsqu'on travaille avec des données déjà structurées comme des informations sur des utilisateurs (âge, taille, couleur de cheveux ...), mais c'est plus difficile avec des images par exemple. Si on travaille sur des images de visages, il faut concevoir des algorithmes capables d'extraire de ces images des vecteurs de caractéristiques décrivant par exemple la taille et la position des yeux, du nez, de la bouche, *etc.* Au contraire, dans une approche d'apprentissage profond, les réseaux de neurones sont aussi utilisés pour l'extraction des caractéristiques, elle est automatique et fait partie du processus d'entraînement. Le plus souvent dans un cadre d'apprentissage supervisé, grâce à une grande quantité de données annotées, le modèle apprend à extraire des caractéristiques de différents niveaux d'abstraction et à déterminer lesquelles sont importantes.

## 2.2 Apprentissage profond

L'apprentissage profond, ou *Deep Learning* en anglais, est un sous-domaine de l'apprentissage automatique exploitant principalement les réseaux de neurones. L'enchaînement de couches de "neurones artificiels" associé à des fonctions d'activation non linéaires permet à ce type d'architecture d'approximer n'importe quelle fonction non linéaire continue. Les "poids" des neurones déterminent les paramètres de ces

fonctions, et sont fixés par un processus d'optimisation d'une fonction de coût et par descente de gradient.

### 2.2.1 Réseaux de neurones

À l'origine, les réseaux de neurones artificiels sont inspirés du cerveau humain. Les neurones artificiels sont une modélisation simplifiée des neurones biologiques. McCulloch et Pitts (1943) sont à l'origine de cette proposition. Ils représentent mathématiquement un neurone comme une fonction qui calcule la somme pondérée d'un vecteur $x$ avec un vecteur $w$, ajoute un terme de biais $b$ puis fait passer le résultat dans une fonction non linéaire $\sigma$ appelée *fonction d'activation* (cf. Equation 2.1).

$$y = \sigma(\sum_i w_i x_i + b) \tag{2.1}$$

Une des propriétés les plus puissantes de cette modélisation est que la combinaison de ces neurones artificiels sous forme de couches qui s'enchaînent (comme sur la Fig. 2.5) permet théoriquement d'approximer n'importe quelle fonction non linéaire continue d'après le *théorème d'approximation universel* (Hornik et al. (1989)).

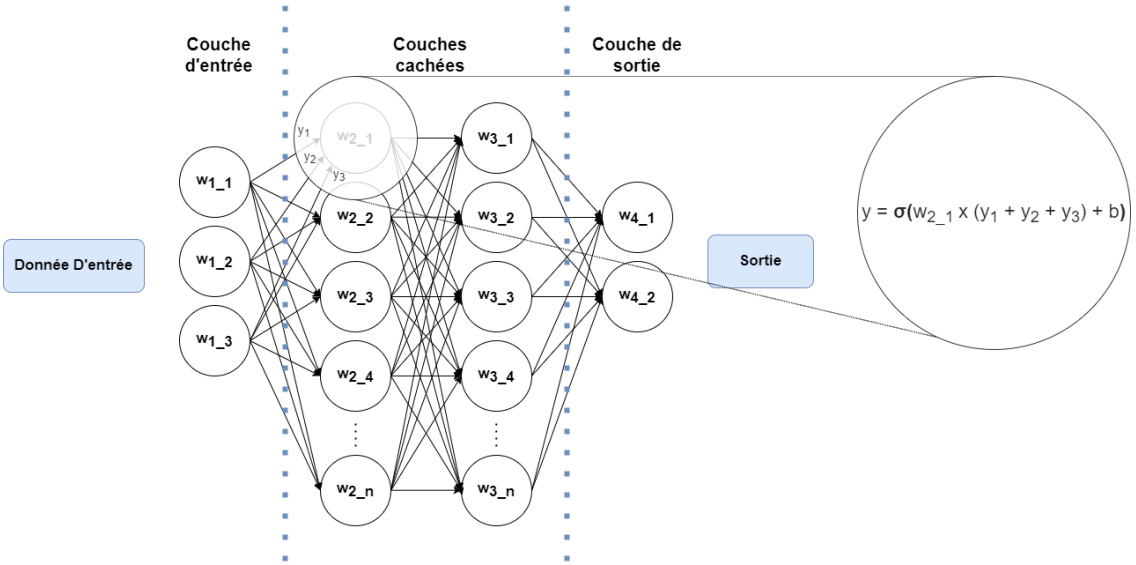



FIGURE 2.5 – Illustration d'un réseau de neurones à deux couches cachées. Chaque neurone possède un paramètre $w$ (pour *weight*, ou "poids" en Français) et calcule l'équation 2.1 à partir de ce paramètre et des valeurs de sorties $y$ des neurones précédents auquel il est connecté.

Il reste à déterminer les valeurs $w$ pour chaque neurone en entraînant le réseau de neurones. Cela est fait itérativement grâce à l'algorithme de *rétropropagation du*

*gradient* (Rumelhart et al. (1986)). Au début de l'entraînement, les poids des neurones sont initialisés aléatoirement. On calcule la sortie du réseau étant donnée une entrée d'entraînement. Cette sortie est comparée à la vérité terrain (qui est la valeur de sortie attendue pour une entrée donnée) *via* la fonction de coût (*loss function*) qui va typiquement renvoyer une valeur grande si le modèle dans son état actuel s'est beaucoup trompé et une valeur petite dans le cas contraire. On peut voir la fonction de coût comme étant une fonction à $n$ variables, autant de variables que de poids dans le réseau, et pour laquelle on cherche la configuration de valeurs de ces variables qui minimise sa valeur de sortie. Pour trouver cette configuration minimisant la fonction de coût, on utilise l'algorithme de *rétropropagation du gradient* qui est un procédé itératif calculant le gradient de la fonction étant donné une configuration, ce qui permet de savoir dans quelle direction aller pour réduire la valeur de la fonction. L'algorithme met à jour les poids des neurones en partant de la dernière couche vers la première couche. Il change la valeur des poids en fonction de l'importance de leur contribution à l'erreur.

Pour arriver à une configuration satisfaisante pour la tâche que l'on souhaite traiter, il est nécessaire d'effectuer ce processus d'entraînement un grand nombre de fois et avec une grande quantité de données. En effet, à chaque passage d'une donnée ou d'un groupe de données dans le réseau, la configuration est un peu modifiée de manière à réduire l'erreur sur ce sous-ensemble de données en particulier. On doit faire passer toutes les données disponibles dans processus, potentiellement un grand nombre de fois au cours de l'entraînement afin que le réseau converge vers une configuration de poids minimisant l'erreur pour toutes les données d'entraînement (Goodfellow et al. (2016)).

L'architecture de réseaux de neurones présentée sur la Figure. 2.5 est de type *feed forward fully connected*, c'est-à-dire qu'il est acyclique et que tous les neurones d'une couche sont connectés avec tous les neurones de la couche suivante. Il existe d'autres architectures qui peuvent être cycliques, comme les réseaux récurrents, partiellement connectés, et même avoir de types de couches particulières, comme dans les **réseaux de neurones convolutifs**.

### 2.2.2 Réseaux de neurones convolutifs

En traitement d'image, un *filtre de convolution* est un filtre qui transforme l'image d'entrée en faisant passer une fenêtre glissante 2D sur l'image, le *noyau de convolution*, calculant une nouvelle valeur du pixel au centre de la zone couverte par le noyau de convolution. Cette nouvelle valeur est calculée en faisant la somme des produits des valeurs des pixels avec les valeurs contenues dans le *noyau de convolution* (cf. Fig. 2.6).

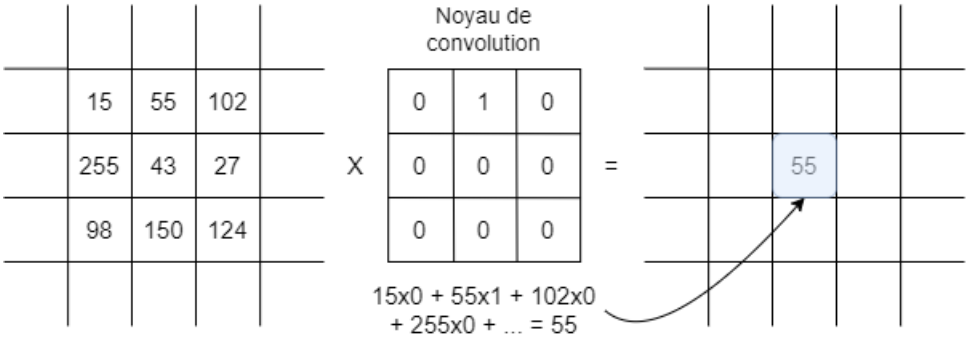



FIGURE 2.6 – Illustration d'un filtre de convolution

En fonction des valeurs, appelées *poids*, présents dans le noyau de convolution, l'image est transformée différemment. Par exemple, avec un noyau de la forme $\begin{pmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{pmatrix}$, les contours horizontaux de l'image sont extraits, avec un noyau de la forme $\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$, l'image est floutée, *etc.*

La force de la convolution 2D réside dans sa capacité à prendre en compte le caractère spatial de l'information contenue dans les images, suivant la taille du noyau de convolution, un voisinage plus ou moins grand autour du point courant est considéré pour le calcul.

La première occurrence d'un réseau de neurones intégrant la notion de convolution peut être associée à l'article de LeCun et al. (1989), dans lequel une architecture comprenant des couches de convolution est utilisée pour reconnaître des codes postaux américains manuscrits. Les mêmes auteurs raffinent ensuite le concept en l'appliquant plus généralement à des documents manuscrits (LeCun et al. (1998)). Depuis, les réseaux de neurones convolutifs sont devenus une des architectures les plus largement utilisées de l'analyse d'image.

Les couches convolutives des réseaux de neurones convolutifs sont constituées de plusieurs filtres de convolution qui transforment l'image de manière à extraire différentes caractéristiques. On appelle les différentes images produites par ces couches des cartes de caractéristiques. Les poids des différents noyaux de convolution sont des paramètres appris lors de l'entraînement du réseau, le modèle apprend donc directement à extraire les caractéristiques importantes pour la tâche voulue, guidé par le processus d'apprentissage basé sur les données.

Ces réseaux sont aujourd'hui organisés en couches successives dans lesquelles s'enchaînent des "couches de convolution" et des réductions de la dimension des cartes de caractéristiques par des opérations de *pooling*. Cela permet aux modèles de progressivement extraire des caractéristiques de plus haut niveau, comme illustré sur la Figure. 2.7. Le *pooling* consiste à sous-échantillonner l'image, ce qui peut se faire

de différentes manières, les plus communes étant le *max pooling* et l'*average-pooling*. Cette opération a pour effet de réduire la taille de l'image, diminuant au passage le nombre de paramètres du modèle et donc le temps nécessaire à son entraînement et son inférence. Mais une de ses propriétés les plus intéressantes est le fait qu'en "agrégeant" les caractéristiques extraites à une échelle donnée dans une échelle plus petite, l'opération de pooling aide aussi à rendre le modèle invariant à de petites translations de l'entrée.

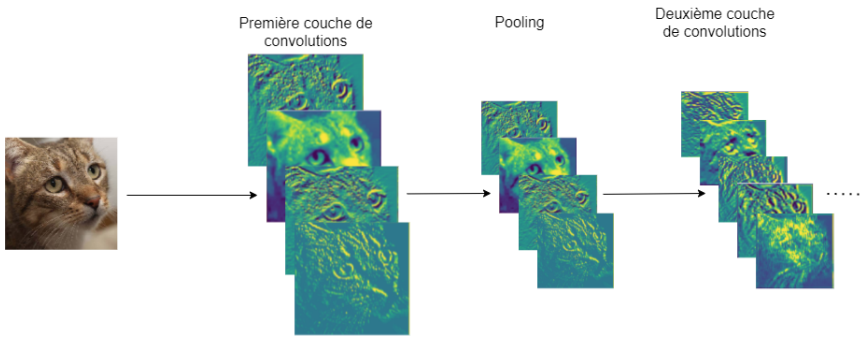



FIGURE 2.7 – Illustration des couches de convolutions d'un réseau de neurones convolutif (fait en utilisant des images provenant de https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2)

Ces enchaînements de couches de convolution servent à l'extraction de caractéristiques pertinentes, mais ne permettent pas de faire des prédictions en elles-mêmes. Les architectures dont le but est la classification d'images rajoutent le plus souvent des couches entièrement connectées juste après les couches de convolution. Elles prennent en entrée les vecteurs de caractéristiques produits par les couches de convolution et calculent des probabilités sur des classes.

Depuis une dizaine d'années, les architectures de réseaux de neurones à base de couches de convolution ont une à une repoussé les limites de l'état de l'art dans le domaine du traitement d'images. Par exemple, pour la classification d'images naturelles, dont le jeu de données de référence est ImageNet [4], la majorité des modèles état de l'art depuis 2013 sont basés sur le principe des couches de convolution (cf. Fig. 2.8).

4. https://www.image-net.org/

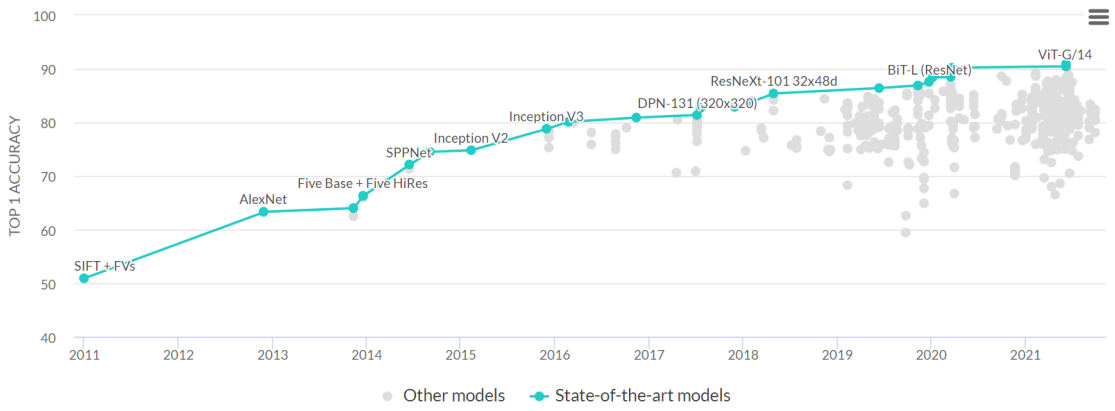



FIGURE 2.8 – Evolution des performances des algorithmes de classification d'images sur le jeu de données *ImageNet* depuis 2011 (source : https://paperswithcode.com/sota/image-classification-on-imagenet)

Ces architectures ont évolué pour devenir de plus en plus profondes, introduisant régulièrement des nouveautés permettant d'améliorer leurs performances. En 2012, *AlexNet* (Krizhevsky et al. (2012)) participe au concours *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)*[5] et réduit de 10.8 points de pourcentage d'erreur l'état de l'art de l'époque, atteignant 63.3% de *top-1 accuracy*. Son architecture contient 5 couches de convolution suivies de trois couches entièrement connectées et la fonction d'activation *ReLU*[6]. De par son succès au *Challenge* ILSVRC, *AlexNet* a eu beaucoup d'influence sur la suite de la recherche en classification d'images. En 2015 est proposée l'architecture *Inception* (Szegedy et al. (2015)). Elle introduit sous la forme d'*Inception Modules* (cf. Fig. 2.9) l'idée d'avoir plusieurs tailles de noyaux de convolution en parallèle pour prendre en compte les variations de la taille et de la localisation des informations dans les images.

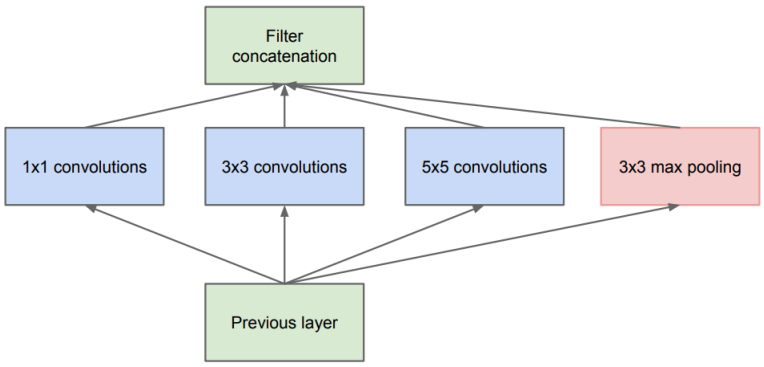



FIGURE 2.9 – Exemple d'*Inception Module*. Source : Szegedy et al. (2015)

5. https://www.image-net.org/challenges/LSVRC/
6. Rectified Linear Unit https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

Toujours en 2015 apparaissent les architectures **VGG** (Liu et Deng (2015)) et **ResNet** (He et al. (2015)).

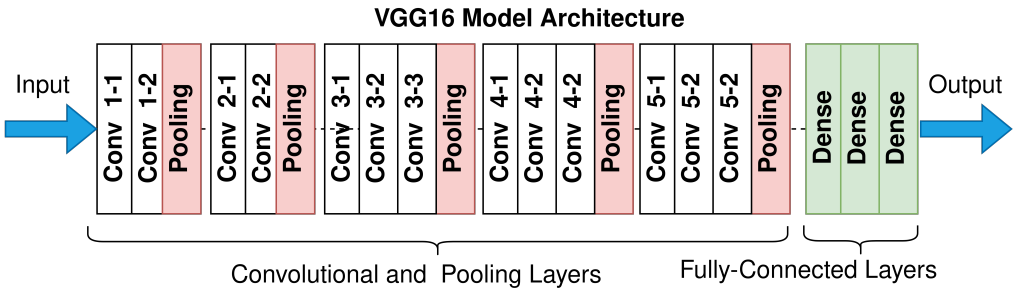



FIGURE 2.10 – Architecture de *VGG16*. Source : https://commons.wikimedia.org/wiki/File:VGG16.png

**VGG** est basé sur *AlexNet*, mais diffère dans son utilisation de petits noyaux de convolution de taille $3 \times 3$ et $1 \times 1$ permettant d'augmenter le nombre de couches de convolutions en limitant l'augmentation du nombre de paramètres qui en découle. Comme on peut le voir sur la Figure. 2.10, la version *VGG16* du modèle est composée d'un enchaînement de blocs constitués de couches de convolutions utilisant la fonction d'activation *ReLu* suivies de *max pooling*. Les deux premiers blocs comportent deux couches de convolution et les trois suivants de trois couches de convolution. Le modèle se termine par trois couches entièrement connectées (aussi appelées *Dense*) utilisant aussi la fonction d'activation *ReLU*.

**ResNet** introduit la notion de *Blocs Résiduels* (cf. Fig. 2.11) qui permet de combattre le problème de la disparition du gradient, ou *The Vanishing Gradient Problem* en anglais. Ce problème commence à apparaître lorsque l'on multiplie le nombre de couches successives dans une architecture de réseaux de neurones profonds utilisant certaines fonctions d'activation. Le gradient de la fonction de coût se rapproche de zéro, ce qui rend l'entraînement de tels réseaux très difficile.

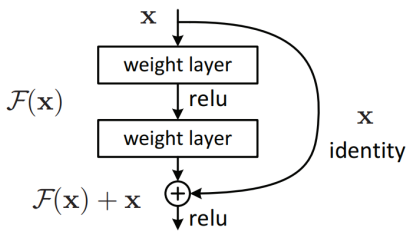



FIGURE 2.11 – Un bloc résiduel. Source : He et al. (2015)

Le principe d'un *bloc résiduel*, aussi appelé *Block identité* (cf. Fig. 2.11) est d'ajouter la valeur du début du bloc à la fin du block, cette valeur ne passant ni par le calcul du poids ni par la fonction d'activation. Les auteurs rapportent que

grâce à ce type de blocs, ils ont été capables de faire converger des architectures de profondeurs considérablement supérieures à précédemment, améliorant par exemple significativement (28% d'amélioration relative) les performances de reconnaissance d'objets sur le jeu de données *COCO*[7]. L'architecture de la version *Resnet50* est illustrée sur la Figure. 2.12.

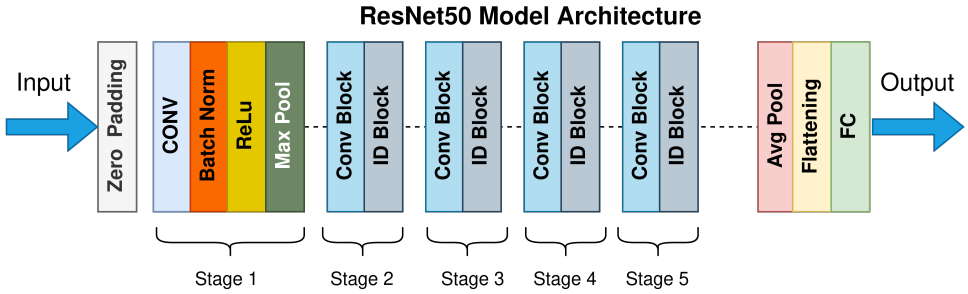



FIGURE 2.12 – Architecture de *Resnet50*. Source : https://commons.wikimedia.org/wiki/File:ResNet50.png

Par la suite, *Inception V2* (Ioffe et Szegedy (2015)) introduit le concept de *Batch Normalization* sur une variante de l'architecture *Inception* et améliore encore l'état de l'art sur la compétition *ILSVRC* avec 74.8% de *top-1 accuracy*. S'en suivent les architectures *Inception v3* (Szegedy et al. (2016)) et *Inception v4* (Szegedy et al. (2017)), dont les évolutions sont détaillées dans un article du blog *Towards data science*[8].

### 2.2.3 Apprentissage par transfert

L'apprentissage par transfert, ou *Transfer Learning* en anglais, est l'ensemble des méthodes permettant de ré-utiliser les connaissances apprises par un modèle pour résoudre un problème en particulier sur un autre problème. Ces méthodes cherchent à améliorer les performances d'un modèle en transférant les informations apprises sur un autre domaine. Bien que cette notion soit générale au *Machine Learning*, elle est aujourd'hui principalement utilisée dans le cadre du *deep learning* (Tan et al. (2018)).

L'apprentissage par transfert est aujourd'hui utilisé pour résoudre le problème de la difficulté d'obtenir les grandes quantités de données annotées nécessaires à l'entraînement de modèles de *deep learning* (Zhuang et al. (2020)). Son principe de fonctionnement est basé sur l'idée que les extracteurs de caractéristiques appris sur une tâche seront probablement utiles sur une autre tâche, ou permettront de plus

7. https://cocodataset.org/
8. https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202

rapidement arriver à des extracteurs pertinents pour notre tâche en ne partant pas de zéro.

Dans le cadre du *Deep Learning*, on peut importer un modèle déjà entraîné sur une tâche, "geler" [9] toutes ou une partie de ses couches et éventuellement rajouter des couches entraînables à la suite des couches "gelées". On lance ensuite le processus de ré-entraînement sur la tâche qui nous intéresse. Dans le cas des architectures à couches convolutives pour le traitement d'images, les couches de convolution, qui sont les extracteurs de caractéristiques, sont suivies par des couches entièrement connectées, qui servent à la classification elle-même à partir des vecteurs de caractéristiques produits par les couches précédentes. Il est alors courant de "geler" toutes les couches convolutives, ou au moins les premières, et de laisser les couches entièrement connectées être ré-apprises en fonction de la nouvelle tâche.

### 2.2.4 Apprentissage *auto-supervisé*

L'apprentissage *auto-supervisé*, ou *self-supervised learning* en anglais, est une forme d'apprentissage à mi-chemin entre l'apprentissage supervisé et l'apprentissage non-supervisé. Il peut être considéré comme non-supervisé de par son utilisation de données non annotées, mais est utilisateur de techniques d'apprentissage supervisées classiques. L'idée de l'apprentissage *auto-supervisé* est d'automatiquement construire un signal de supervision pouvant être utilisé pour faire de l'apprentissage supervisé. Cela se traduit le plus souvent par une annotation automatique des données en utilisant les informations intrinsèques des données elles-mêmes. L'objectif est d'apprendre de manière supervisée sur une *tâche-prétexte* dont les annotations peuvent être générées automatiquement à partir des données. La *tâche-prétexte* doit être conçue de manière à "inciter" le modèle à apprendre des connaissances utiles pour résoudre la *tâche-objectif*, qui est l'objectif final.

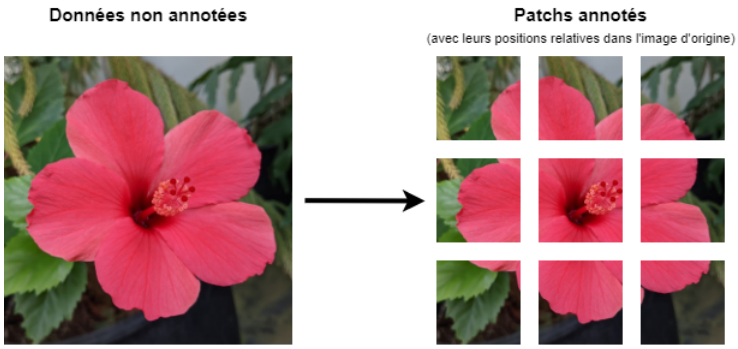



FIGURE 2.13 – Création automatique d'annotations pour la prédiction de positions relatives de régions d'images.

9. "Geler" une couche veut ici dire qu'on spécifie à l'optimiseur que les poids de ces couches ne doivent pas être affectés pendant l'entraînement.

Un exemple de tâche se prêtant bien à l'apprentissage *auto-supervisé* est la prédiction du positionnement relatif de régions d'images (Noroozi et Favaro (2016)). La vérité terrain peut être construite automatiquement en découpant des images en conservant l'information de leurs positions relatives (cf. Fig. 2.13). Un modèle peut ainsi être entraîné à prédire la position relative de deux patchs provenant de la même image.

Une autre tâche pour laquelle il semble naturel d'utiliser une approche *auto-supervisée* est la coloration d'images (Deshpande et al. (2015) Iizuka et al. (2016) Larsson et al. (2016)). En effet, pour produire des données d'entraînement, il suffit de convertir automatiquement une grande base de données d'images couleur en niveaux de gris. On peut ensuite entraîner un modèle à coloriser des images de manière supervisée à partir des images en niveaux de gris et avec la vérité terrain des images couleur originales. On peut aussi citer le domaine de la *super-résolution* (Nguyen et al. (2021a), Zhao et al. (2020)), dans lequel on cherche à augmenter la résolution des images, ou il suffit encore une fois de produire automatiquement des images de plus basse résolution à partir d'une grande base de données d'images haute résolution. On entraîne ensuite un modèle à reconstruire des images haute résolution de manière supervisée.L'étude Jing et Tian (2020) propose une catégorisation de différentes tâches prétextes en fonction de leurs domaines, comme on peut le voir sur la Figure. 2.14. Les différents domaines illustrés sur la Figure. 2.14 peuvent être utilisateurs de méthodes différentes, mais ont en commun d'avoir la possibilité d'apprendre sur des *tâches prétexte* conçues à partir de données non annotées.

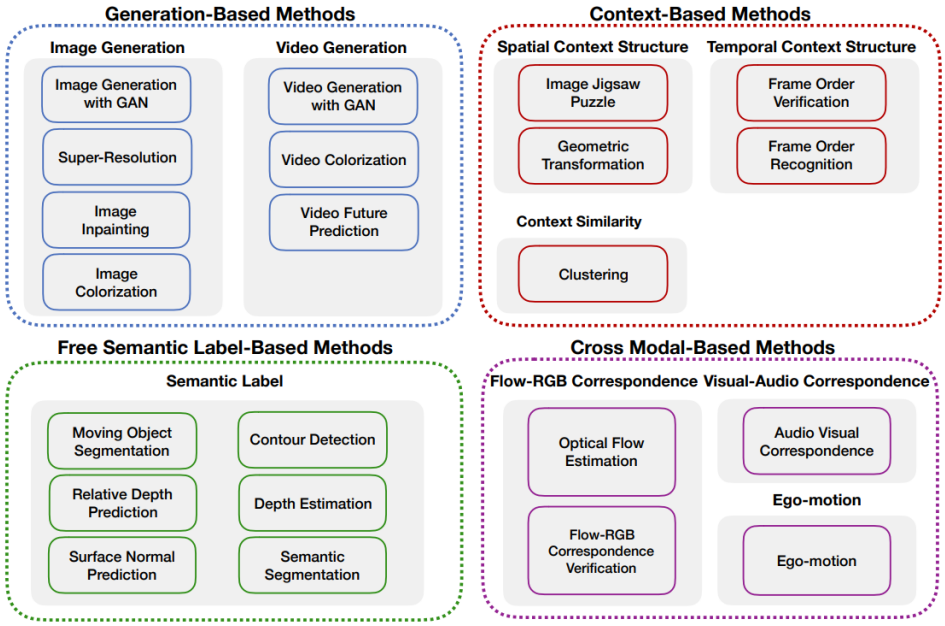



FIGURE 2.14 – Exemples de domaines d'applications dans lesquels une approche *auto-supervisée* est possible. Source : Jing et Tian (2020)

## 2.3 Bonnes pratiques pour l'entraînement l'évaluation de modèles

Il existe un certain nombre de bonnes pratiques pour conduire des expériences utilisant des algorithmes de *Deep Learning*. Il est nécessaire de faire attention à certaines pratiques lors des phases de préparation des données, d'entraînement des modèles et d'évaluation des résultats. Les éléments présentés ci-dessous sont une synthèse de quelques chapitres du livre de référence *Deep Learning* de *Ian Goodfellow*, *Yoshua Bengio* et *Aaron Courville* (Goodfellow et al. (2016)) qui traite de cette question.

Un des écueils les plus courants est le **sur-apprentissage**, ou *overfitting* en anglais. Ce phénomène apparaît lorsque le modèle qu'on a entraîné présente de très bonnes performances sur les données d'entraînement et de mauvaises performances sur de nouvelles données qui n'ont jamais été utilisées lors de l'entraînement. Le modèle a appris le jeu de données au lieu de **généraliser**, c'est-à-dire de découvrir et d'apprendre les informations sous-jacentes permettant une compréhension générale du problème représenté dans les données d'entraînement (cf. Fig. 2.15).

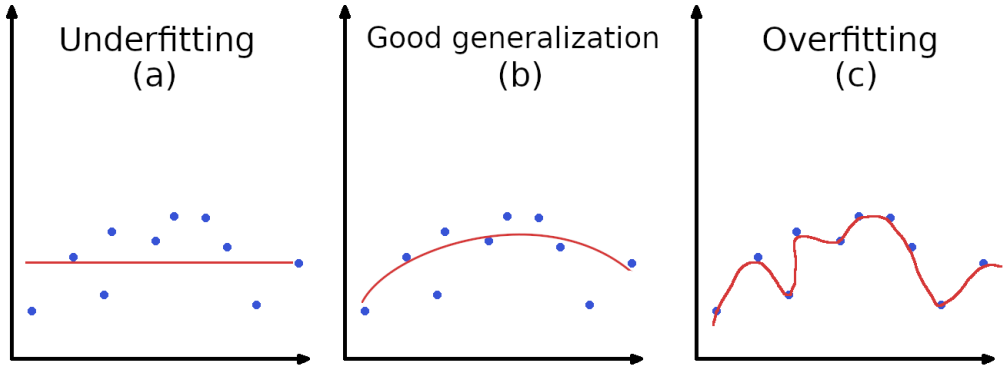



FIGURE 2.15 – Illustration du sous-apprentissage et du sur-apprentissage

Pour identifier ce comportement, la pratique qui fait consensus est de diviser nos données en un jeu d'entraînement et un jeu de test. On peut même aller jusqu'à constituer un jeu de test avec des données acquises différemment du jeu d'entraînement. Les données d'entraînement ne servent que pour entraîner le modèle et ne sont jamais utilisées dans la phase d'évaluation. Inversement, les données de test sont mises de côté pour la phase d'évaluation et ne sont jamais utilisées pour l'entraînement du modèle. Lors de la constitution en amont de ces deux jeux de données, il est important de faire en sorte que le jeu de test, dont la taille est souvent d'environ 10% de la quantité totale de données, soit représentatif des données disponibles. Ainsi, si on observe une différence trop importante entre les performances du modèle sur les données d'entraînement et celles sur les données de test, on peut conclure

que le modèle à sur-appris les données d'entraînement. Concrètement, ce sont les performances relevées sur le jeu de test qui feront foi de la qualité du modèle.

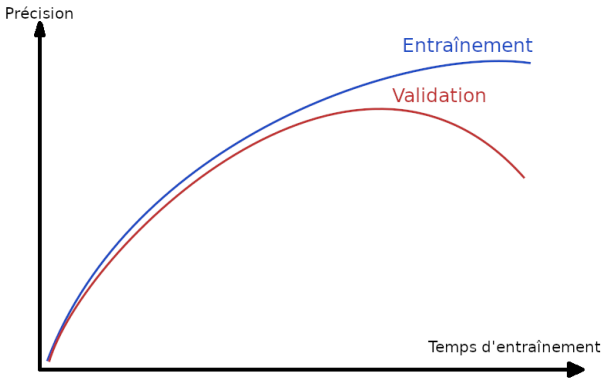



FIGURE 2.16 – Courbes de précision d'entraînement et de validation symptomatiques d'un phénomène de sur-apprentissage

Il est même courant de diviser les données en trois jeux distincts, sans intersection entre eux : entraînement, validation et test. Le jeu de validation sert à surveiller l'éventuel sur-apprentissage lors de l'entraînement. Ses données ne sont pas utilisées pour entraîner le modèle, elles ne contribuent pas à la mise à jour des poids du réseau de neurones par exemple, mais on mesure périodiquement les performances du modèle pendant l'entraînement en calculant des résultats sur le jeu de validation. On peut simplement calculer la fonction de coût de la même manière que sur les données d'entraînement ou utiliser d'autres métriques comme la précision. Cela permet d'afficher des courbes très utiles pour surveiller le processus d'apprentissage d'un modèle, comme illustré sur la Figure. 2.16.

Dans le cas d'un jeu de données déséquilibré, c'est-à-dire qui diffère fortement dans son nombre d'éléments par classe, il est important de faire en sorte que les différents jeux contiennent des proportions similaires d'éléments par classe. On notera que ce déséquilibre peut aussi induire en erreur si les métriques d'évaluation choisies ne sont pas adaptées. Nous discuterons de ce problème plus en détail dans la Section. 3.2.5.

Le problème du sur-apprentissage est souvent dû à une "capacité" du modèle trop importante par rapport à la quantité de données. La "capacité" d'un modèle peut être définie comme son aptitude à représenter variété de fonctions (Goodfellow et al. (2016)). Plus le modèle est complexe, en termes de quantité de neurones par exemple, plus il est capable d'approximer une grande variété de fonctions. Si la "capacité" du modèle est trop grande par rapport à l'entropie [10] des données, alors après un grand nombre d'itérations d'entraînement, le modèle est capable de retenir les données plutôt que de généraliser.

10. Au sens de l'entropie de Shannon : fr.wikipedia.org/wiki/Entropie_de_Shannon

Pour se prévenir du phénomène de sur-apprentissage, on peut utiliser plus de données, ou réduire la capacité du modèle. On peut aussi employer des techniques dites de "régularisation", permettant d'améliorer les capacités de généralisation des algorithmes d'apprentissage (Goodfellow et al. (2016)) en pénalisant les paramètres de l'algorithme d'apprentissage en réduisant leurs magnitudes. Une explication donnée par *Andrew Ng* dans son cours de *Machine Learning* à l'Université de Stanford [11] pour en avoir une intuition est la suivante. Si on reprend l'exemple de la Figure. 2.15, la fonction représentée par la courbe "(c)" a plus de paramètres que la fonction représentée par la courbe "(b)". Disons que cette dernière est quadratique, un polynôme de degré 2, et que la première est un polynôme de degré 4, comme représenté dans les équations 2.2 et 2.3.

$$(b) = \theta_1 x^2 + \theta_2 x + \theta_3 \tag{2.2}$$

$$(c) = \theta_1 x^4 + \theta_2 x^3 + \theta_3 x^2 + \theta_4 x + \theta_5 \tag{2.3}$$

On voit qu'en réduisant l'amplitude des paramètres $\theta_1$ et $\theta_2$, on se rapproche d'une fonction quadratique, réduisant la complexité de la fonction initiale. On diminue ainsi l'impact de ces paramètres sans pour autant les supprimer complètement.

Une autre technique aujourd'hui classique est le *Dropout*, introduit par Srivastava et al. (2014). Elle consiste à inhiber aléatoirement des neurones pendant le processus d'entraînement. On définit à l'avance un hyper-paramètre correspondant à la probabilité pour un neurone d'être inhibé. À 0.5, environ 50% des neurones seront désactivés à chaque itération de données d'entraînement, et ces neurones seront différents à chaque fois. On peut contrôler sur quelles couches le *dropout* sera actif, ainsi que le degré de régularisation souhaité. Cette méthode agit à la fois comme une méthode de régularisation et comme une façon de robustifier le modèle. Elle réduit la possibilité de sur-apprentissage, car à chaque itération on entraîne un modèle ayant une capacité plus faible, mais aussi, car on entraîne une multitude de sous-réseaux dont l'éventuel sur-apprentissage individuel sera différent d'un réseau à l'autre et sera en quelque sorte moyenné sur le réseau final (Nielsen (2015)). Cette méthode possède les avantages d'être très simple à implémenter et à comprendre, d'être très peu coûteuse en calculs supplémentaires et de bien fonctionner sur n'importe quel type d'architecture de réseaux de neurones (Goodfellow et al. (2016)).

Il peut aussi être une bonne idée de faire de l'augmentation de données pour des raisons de régularisation. Cela consiste à multiplier la quantité de données disponibles en appliquant des transformations à une partie ou tout le jeu de données d'entraînement. Des détails sur ce procédé sont donnés dans la Section. 4.4.

11. https://www.youtube.com/watch?v=KvtGD37Rm5I

## 2.4 Apprentissage de métriques

Nous avons établi dans les sections précédentes les bases de l'apprentissage automatique et de l'apprentissage profond. Nous nous intéressons dans cette section à l'apprentissage de métriques, ou *Metric Learning* en anglais, d'abord d'un point de vue *Machine Learning* puis d'un point de vue *Deep Learning* avec l'apprentissage profond de métriques, ou *Deep Metric Learning (DML)* en anglais.

On définit premièrement une métrique de distance comme étant une fonction calculant la distance séparant deux points dans un espace à $n$ dimensions. Une métrique de distance $d(x, y)$ doit satisfaire les propriétés suivantes :

**Elle respecte le *Principe d'identité des indiscernables***

$$d(x, y) = 0 \iff x = y$$

**Elle est *Symétrique***

$$d(x, y) = d(y, x)$$

**Elle satisfait *L'inégalité triangulaire***

$$d(x, y) \leq d(x, z) + d(z, y)$$

Comme expliqué dans le livre Bellet et al. (2015), le *Metric Learning* consiste à apprendre automatiquement comment mesurer correctement une similarité ou une distance à partir d'informations sur les données de la forme "$x$ est similaire à $y$" ou "$x$ est plus similaire à $y$ que $z$". De manière générale, en *Metric Learning* on cherche à représenter les vecteurs de caractéristiques extraits des données dans un espace dans lequel il y a du sens de calculer une distance entre ces vecteurs. En effet, en représentant nos données sous la forme de vecteurs de caractéristiques numériques, on pourrait se contenter d'utiliser une distance euclidienne, mais elle mène souvent à des résultats sous-optimaux, pour la raison principale qu'elle ne prend pas en compte à quel point les différentes caractéristiques sont plus ou moins importantes.

Les auteurs de Bellet et al. (2015) donnent l'exemple d'une base de données d'images de visages. Dans le cas où on cherche à faire de l'identification de visages, on peut considérer que des caractéristiques importantes sont la couleur des cheveux, la forme du visage, la distance entre les yeux, etc. Par contre, si l'on cherche à déterminer l'expression faciale, les caractéristiques qui seront retenues seront plutôt par exemple la position des sourcils, de la bouche ou des pommettes. Il faut noter que ces caractéristiques peuvent varier de façon corrélée les unes par rapport aux autres. Il est donc très difficile, en pratique, de définir la mesure de similarité optimale pour une tâche précise.

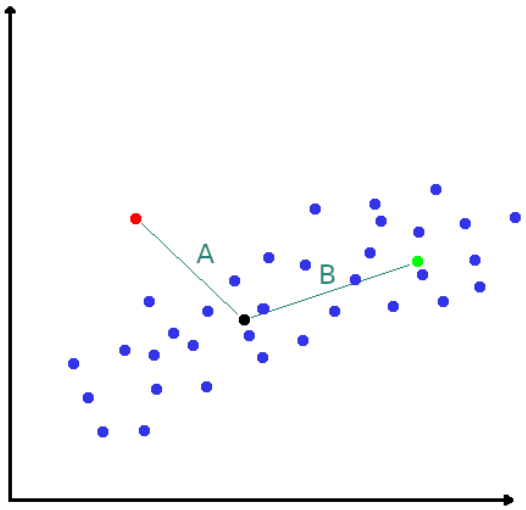



FIGURE 2.17 – Illustration de la distance de Mahalanobis

Les techniques de *Metric Learning* classiques, qui empruntent beaucoup aux techniques statistiques, utilisent entre autres les distances de *Mahalanobis* (Mahalanobis (1936)), de *Bhattacharyya* (Bhattacharyya (1943)) ou de *Kullback-Leibler* (Kullback et Leibler (1951)). Par exemple, la distance de *Mahalanobis* prend en compte la *covariance* des caractéristiques dans son calcul. Sur l'exemple illustré sur la Figure. 2.17, la distance euclidienne entre les points rouge et noir $A$ est plus petite que la distance $B$, mais les caractéristiques ont une covariance et le point vert semble mieux suivre la tendance, ce qui incite à vouloir le classifier comme étant plus proche du point noir que le point rouge. L'algorithme de la distance de *Mahalanobis* transforme linéairement l'espace de manière à éliminer la covariance entre les caractéristiques et calcule finalement une distance euclidienne sur cet espace transformé. C'est d'une certaine manière un algorithme d'apprentissage, car les paramètres de la transformation sont calculés en fonction des caractéristiques extraites des données.

### 2.4.1 Apprentissage profond de métriques

L'apprentissage profond de métriques a le même objectif que l'apprentissage de métriques, mais exploite les méthodes de l'apprentissage profond pour apprendre à calculer des métriques de distances à partir des données, de leurs annotations et d'une fonction de coût.

Cette famille de méthodes est typiquement utilisée dans le domaine de la re-identification de personnes (*Person re-identification* ou *ReID*) (Masi et al. (2018)), où l'on cherche à déterminer si deux photos de visages correspondent à la même

personne – avec plusieurs photos de la même personne sous différents angles, conditions lumineuses, appareils de capture, *etc.* Le principe est d'optimiser le modèle afin d'avoir une grande distance entre deux photos de personnes différentes et une petite distance entre deux photos de la même personne. Les mêmes principes sont par exemple utilisés dans le domaine de la vérification de signatures (Stauffer et al. (2021)).

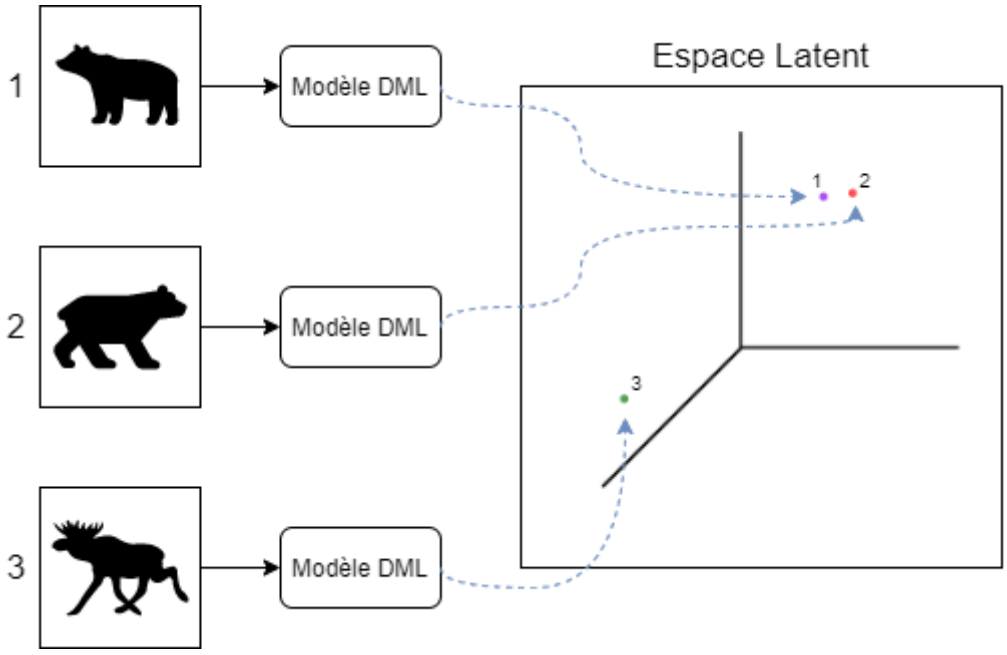



FIGURE 2.18 – Représentation du fonctionnement d'un algorithme de *Deep Metric Learning*

Pour résoudre ces problèmes, l'état de l'art utilise désormais comme dans beaucoup de domaines le *Deep Learning* (KAYA (2019)). L'approche la plus commune est une architecture de réseau de type *siamois* ou *triplet*. Ce sont des architectures de réseaux de neurones à plusieurs *branches* dont le but est d'obtenir une représentation des données ayant des propriétés qui nous intéressent dans un *espace latent*. Un *espace latent* est un espace multi-dimensionnel abstrait dans lequel existent des caractéristiques ne pouvant pas être interprétées directement, mais qui encode une représentation interne des caractéristiques qui a du sens. Dans le cas des réseaux *siamois* ou *triplet*, il est important que les *branches* effectuent exactement les mêmes opérations pour que l'on puisse appeler cela une métrique, car le modèle doit respecter les propriétés des métriques énoncées dans la section 2.4.

En *Deep Metric Learning*, on ne cherche pas à classifier nos données directement – attribuer une étiquette (label) aux exemples – mais on peut se servir des distances élément à élément pour construire des clusters et finalement arriver à une forme de classification. La différence principale avec la classification *classique* est que l'on ne

connaît pas nécessairement à l'avance le nombre de classes dans lesquelles existent les données. De plus, les méthodes utilisées sont adaptées à des topologies de données dans lesquelles il existe souvent un grand nombre de classes et peu d'exemples de chaque classe (beaucoup de personnes différentes, quelques photos de chaque personne). Par exemple, la base de données *Labeled Faces In The Wild*[12] est composée 5749 classes et de 13233 images, pour une moyenne d'environ 2.3 images par classes. Généralement, c'est plutôt l'inverse. Par exemple, la base de données *ImageNet* est composée de 20000 classes et de 14 millions d'images pour une moyenne d'environ 700 images par classe.

L'article Horiguchi et al. (2016) comparant la pertinence d'utiliser une approche *Deep Metric Learning* plutôt qu'un classifieur de type *softmax* classique pour l'extraction de caractéristiques conclut que dans le cas d'un très grand nombre de classes contenant peu d'exemples, l'approche *Deep Metric Learning* est la plus pertinente.

Concrètement, le *Deep Metric Learning* cherche la meilleure manière de transformer la donnée requête de manière à la représenter dans un espace latent commun avec les données qu'on veut comparer (KAYA (2019)). Ce processus de transformation est guidé de façon à ce que deux entrées considérées comme "proches" ou "similaires" soient projetées proches l'une de l'autre dans l'espace latent et inversement pour deux entrées considérées "éloignées", comme on peut le voir sur la Figure. 2.18. Ainsi, on peut mesurer la distance entre les projections pour déterminer une métrique de similarité.

La métrique de distance est ainsi apprise en fonction des données en entrée. Ce sont par exemple les annotations qui déterminent la notion de "similarité" dans le cas ou le jeu d'entraînement est constitué de classes d'images, on considérera comme "similaire" deux images provenant de la même classe et "dissimilaire" deux images provenant de différentes classes. On entraîne ainsi un modèle unique qui projette les différentes entrées dans l'espace latent. Pour y arriver, la façon la plus commune est d'utiliser une architecture de réseaux de neurones appelée *Réseaux Siamois*.

### 2.4.2 Réseaux Siamois

On appelle *Réseau Siamois* un réseau de neurones composé de deux branches identiques dont les poids sont partagés lors des phases d'entraînement et d'inférence. Une branche est constituée d'un ensemble de couches de neurones dont l'organisation dépend du cas d'application et du type de données traitées. Dans sa forme la plus élémentaire, un réseau siamois prend deux données en entrée (une par branche) et calcule le résultat de chaque branche indépendamment, généralement sous la forme d'un vecteur *1-D*. Une distance est ensuite calculée entre les deux vecteurs, ce qui constitue la sortie finale du réseau siamois (cf. Fig. 2.19).

12. http://vis-www.cs.umass.edu/lfw/

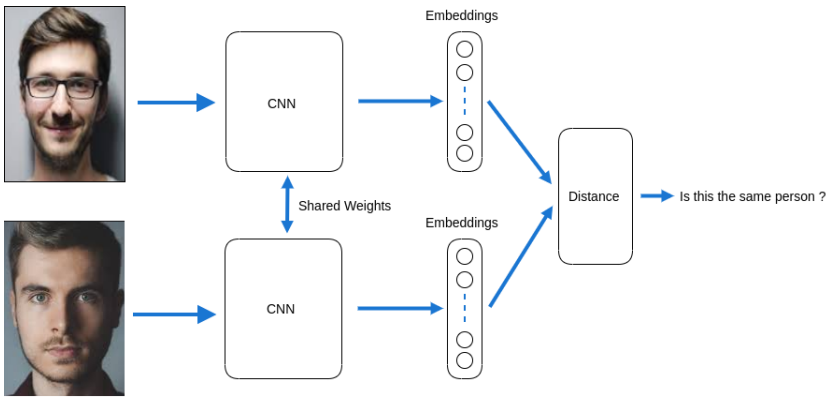



FIGURE 2.19 – Exemple d'architecture de réseau siamois. Étant données deux photos de visages en entrée des deux branches, on cherche à déterminer si ce sont des photos de la même personne.

Les réseaux siamois ont été introduits par Bromley et al. (1993) comme approche pour résoudre le problème de la vérification de signatures (*Signature Verification*). Hadsell et al. (2006) ont ensuite proposé une nouvelle fonction de coût pour l'entraînement de réseaux siamois, la *Contrastive Loss* :

$$L(Y, \vec{X_1}, \vec{X_2}) = (1 - Y)\frac{1}{2}(D(\vec{X_1}, \vec{X_2}))^2 + (Y)\frac{1}{2}\{max(0, m - D(\vec{X_1}, \vec{X_2}))\}^2$$

Avec $\vec{X_1}$ et $\vec{X_2}$ les vecteurs de caractéristiques en sortie de chaque branche du réseau siamois, $Y$ la vérité terrain $Y = 0$ si les deux entrées sont similaires et $Y = 1$ si les deux entrées sont dissimilaires. $D$ est une fonction calculant une distance entre les deux vecteurs. $m$ est un paramètre de "marge" fixé.

Cette fonction de coût renvoie une grande valeur si la distance entre les deux vecteurs est grande alors que les entrées sont similaires ou si la distance entre les deux vecteurs est petite alors que les entrées sont dissimilaires. Elle renvoie une petite valeur dans les deux cas inverses. Elle renvoie 0 dans le cas où les entrées sont dissimilaires, mais que la distance entre les deux vecteurs est plus grande que la marge $m$. Tout cela a pour effet de forcer le modèle à se rapprocher d'une configuration de poids dans laquelle deux entrées similaires seront projetées proches dans l'espace latent et deux entrées dissimilaires seront projetées éloignées dans l'espace latent.

### 2.4.3 Réseaux Triplet

Une évolution des réseaux siamois a été de considérer des triplets d'entrées plutôt que des paires. On peut ainsi définir que l'entrée $x$ est plus proche de l'entrée $y$ que l'entrée $z$. De cette idée est née la *Triplet Loss*, proposée par Schroff et al. (2015) :

$$L = max(D(a, p) - D(a, n) + m, 0) \tag{2.4}$$

Avec $D$ une fonction de distance comme définie dans 2.4.2, $a$, $p$ et $n$ trois vecteurs de caractéristiques en sortie de chaque branche du réseau triplet et $m$ un paramètre de marge fixé. Lors de l'entraînement, on forme des triplets $(a, p, n)$, $a$ est "l'ancre", $p$ est un exemple similaire à $a$ et $n$ est un exemple dissimilaire à $a$. L'idée de cette fonction de coût est de rapprocher l'exemple positif de l'ancre et d'éloigner l'exemple négatif de l'ancre, comme illustré sur la Figure. 2.20. La marge $m$ définit un seuil à partir duquel on considère que l'exemple positif est suffisamment plus proche de l'ancre que l'exemple négatif. En effet, il suffit que $D(a, p) - D(a, n)$ soit inférieur à $m$ pour que la fonction retourne 0, indiquant en quelque sorte qu'il n'y a aucune erreur dans la prédiction.

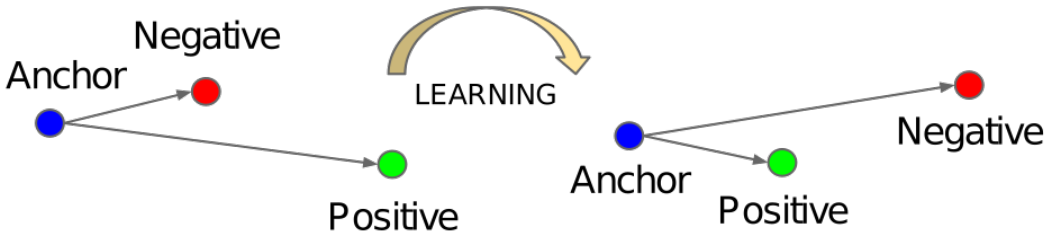



FIGURE 2.20 – Illustration de la triplet loss (Source de l'image Schroff et al. (2015))

De nombreuses autres fonctions de coût ont été développées comme illustré sur la Figure. 2.21. Par exemple, la *quadruplet loss* (Chen et al. (2017)) cherche à augmenter la variabilité inter-classes et à diminuer la variabilité intra-classe des caractéristiques. La *center loss* (Wen et al. (2016)) ajoute un terme de régularisation à la classique *softmax loss* afin de pousser les caractéristiques à se regrouper vers le centre des classes dans l'espace de représentation. On peut remarquer que le développement de ces fonctions de coût s'est principalement fait dans le cadre de la reconnaissance/discrimination de visages, comme *SphereFace* (Liu et al. (2017)), *CosFace* (Wang et al. (2018)) ou encore *ArcFace* (Deng et al. (2019)). D'autres exemples de fonctions de coût ayant les mêmes objectifs que celles présentées précédemment sont décrits dans Kha Vu (2021).

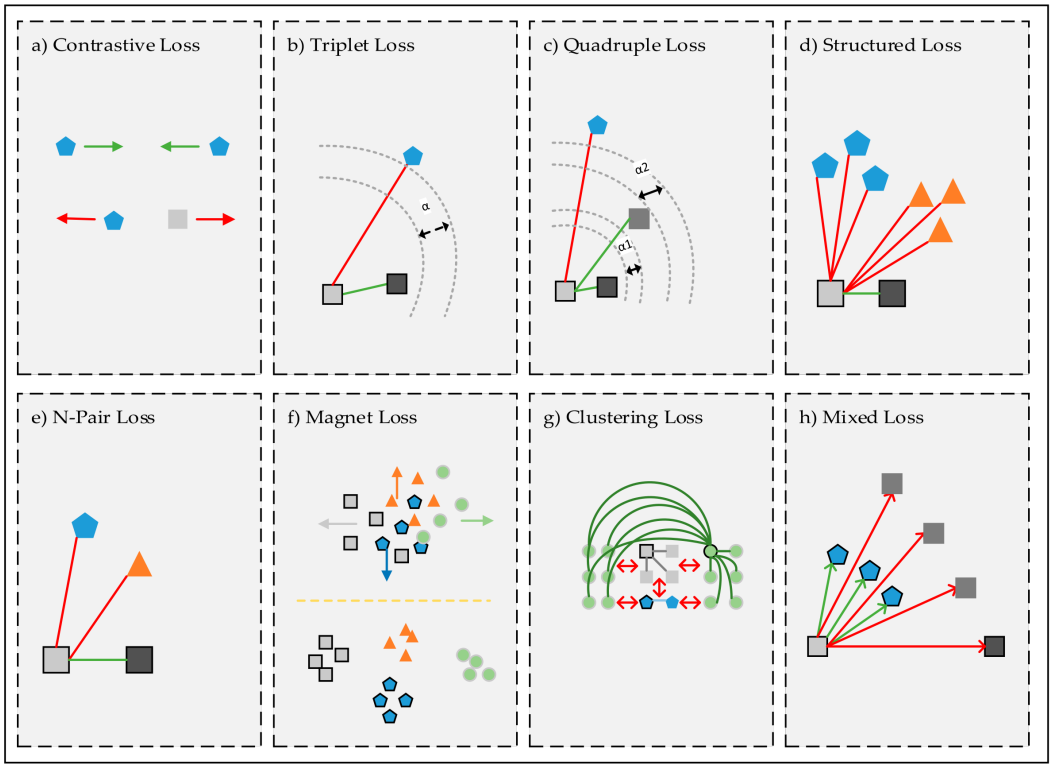



FIGURE 2.21 – Illustrations des différentes losses (Source de l'image KAYA (2019))

### 2.4.4 "Minage" de triplets

L'article Schroff et al. (2015) propose aussi une approche pour optimiser l'entraînement des réseaux triplets utilisant la *Triplet Loss*, mais le concept peut s'appliquer de manière plus générale : le "minage de triplets", ou *Triplet Mining* en anglais. Les auteurs font le constat qu'une grande partie des triplets qu'il est possible de construire sont "faciles", c'est-à-dire que l'élément positif est déjà plus proche de l'ancre que l'élément négatif (en prenant en compte la marge $m$). En d'autres termes, si un triplet $(a, p, n)$ satisfait la condition énoncée dans l'équation. 2.5, la *Triplet Loss* renvoie 0.

$$D(a, p) + m < D(a, n) \tag{2.5}$$

Ce triplet ne contribuerait donc pas à la progression de l'entraînement, ralentissant la convergence. Les auteurs préconisent donc de sélectionner des triplets qui violent la condition de l'équation 2.5. Comme illustré sur la Figure. 2.22, il existe trois types de triplets, les faciles, les semi-difficiles et les difficiles. Les triplets semi-difficiles se trouvent "dans" la marge, c'est-à-dire que l'élément négatif est à une distance plus grande de l'ancre que l'élément positif, mais la différence de distance est suffisamment petite pour se trouver dans la marge $m$. Les triplets difficiles, quant

à eux, correspondent à des triplets dans lesquels l'exemple négatif est plus proche de l'ancre que l'exemple positif.

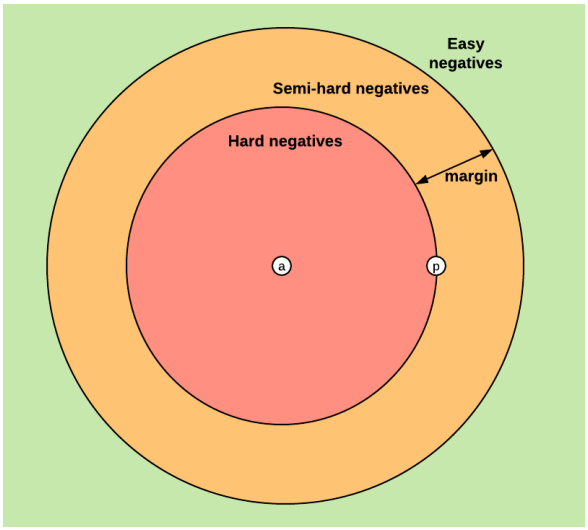



FIGURE 2.22 – Illustrations des différents types de triplets qu'il est possible de construire pour faire du *Triplet Mining*. Source : https://omoindrot.github.io/triplet-loss

Il est peu envisageable d'effectuer les calculs en construisant tous les triplets possibles sur tout le jeu de données à chaque étape d'entraînement (*Offline Mining*), cela augmenterait massivement les temps d'entraînement. Les auteurs proposent plutôt de sélectionner les triplets intéressants à la volée, lors de la constitution des batchs d'entraînement (*Online Mining*). On construit chaque batch avec $n$ exemples d'entraînement, puis on forme tous les triplets valides possibles, c'est-à-dire les triplets contenant deux "positifs" et un "négatif". On calcule ensuite les distances sur ces triplets et on peut ainsi sélectionner ceux qui nous intéressent pour l'entraînement, et même panacher le contenu du batch avec des exemples faciles, semi-difficiles et difficiles. Les vecteurs de caractéristiques ont déjà été calculés pour calculer les distances, on se contente de ne donner à l'optimiseur que les triplets choisis pour la rétropropagation.

## Conclusion

Dans ce chapitre, nous avons exposé les éléments d'apprentissage automatique et d'apprentissage profond nécessaires à la compréhension technique de ce manuscrit. Nous avons décrit l'évolution des architectures de réseaux de neurones convolutifs, l'intérêt de l'apprentissage par transfert ainsi que les principes de l'apprentissage *auto-supervisé*. Après avoir passé en revue les principales bonnes pratiques pour l'entraînement et l'évaluation de modèles d'apprentissage profond, nous nous sommes concentrés sur l'apprentissage profond de métriques, qui est au cœur des proposi-

tions de ces travaux de thèse, et plus particulièrement l'utilisation de réseaux de neurones siamois.

Dans le chapitre suivant, nous présentons le modèle que nous avons retenu et décrivons les différentes étapes que nous avons mises en place pour atteindre notre objectif, qui est de calculer des distances entre des fragments de papyrus.

# Chapitre 3

# Réseaux Siamois pour la suggestion d'assemblages de fragments de documents anciens

**Sommaire**

# Introduction

L'objectif de ces travaux de thèse est de proposer des méthodes pour faire de la suggestion d'appairages de fragments de documents. Ces suggestions ont vocation à être utiles à des chercheurs en humanités numériques ayant besoin de reconstruire ces documents fragmentaires pour les étudier. Pour cela, nous proposons d'avoir recours aux techniques d'apprentissage automatique de distances afin de calculer des similarités entre fragments. Parmi les approches présentées dans le chapitre 2, nous choisissons de nous concentrer sur les réseaux de neurones siamois.

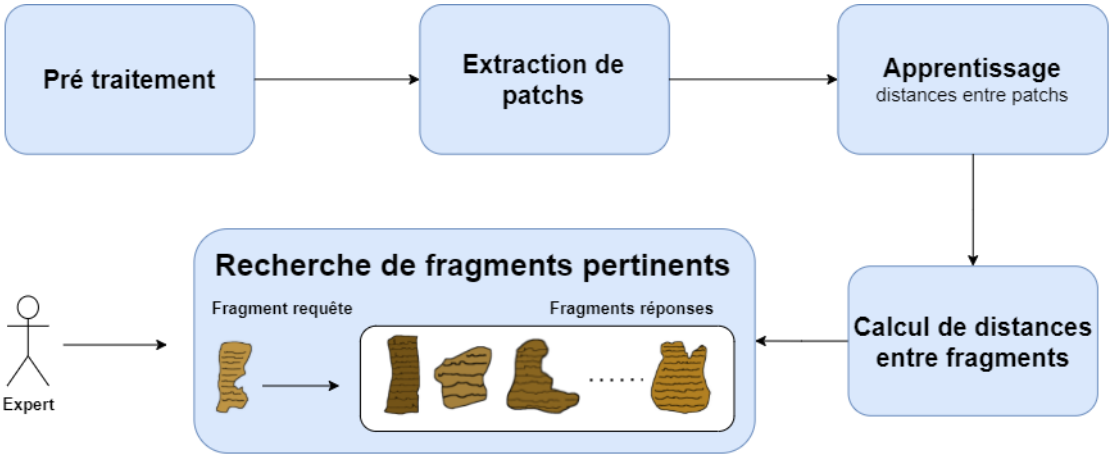



FIGURE 3.1 – Vue d'ensemble du processus proposé pour obtenir des suggestions d'appairage de fragments. Après une phase de pré-traitement et une extraction contrôlée de patchs, des modèles siamois sont entraînés pour apprendre à calculer des distances entre patchs. À partir de ces distances, on calcule des distances entre fragments. On peut ainsi pour un fragment requête donné, proposer une liste classée de fragments réponses à un expert.

Nous décrivons premièrement dans la section 3.2 la méthodologie que nous développons pour proposer des suggestions d'assemblages de fragments de documents ainsi que le protocole d'évaluation et ses métriques. Cette méthodologie est illustrée sur la Figure. 3.1.

Dans la section 3.3 nous exposons les deux bases de données que nous avons utilisées pour conduire nos expériences, leurs spécificités et les éventuels pré-traitements qui ont été nécessaires à leur utilisation. Une de nos contributions a été de construire une base de données d'entraînement significative à partir de la collection de papyrus de l'Université du Michigan que nous avons mis à disposition des chercheurs en traitement et analyse d'image [1].

Ensuite, dans la section 3.4, nous conduisons des expériences préliminaires visant à déterminer si l'architecture choisie est capable de correctement déterminer si deux

1. https://zenodo.org/record/4680207

patchs proviennent du même fragment. Nous explorons aussi l'impact de la présence de texte dans les patchs sur les performances du modèle.

Enfin, nous appliquons la méthodologie exposée dans la section 3.2 à nos deux bases de données dans la section 3.5. Nous expérimentons sur deux architectures de réseaux de neurones convolutifs, *VGG16 et Resnet50* en plus de l'architecture *Papy-S-Net* que nous avons proposée pour les expériences préliminaires de la section 3.4.

Les travaux présentés dans ce chapitre ont donné lieu à deux publications:

▷ [Pirrone et al. (2019)] Antoine Pirrone, Marie Beurton-Aimar, Nicholas Journet. **Papy-S-Net: A Siamese Network to match papyrus fragments**, *In Proceedings of the 5th International Workshop on Historical Documents Imaging and Processing 2019*

▷ [Pirrone et al. (2021)] Antoine Pirrone, Marie Beurton-Aimar, Nicholas Journet **Self-supervised deep metric learning for ancient papyrus fragments retrieval**, *International Journal on Document Analysis and Recognition (IJDAR 2021)*

## 3.1 État de l'art sur la reconstruction de documents

Il existe différents types de supports fragmentaires sur lesquels une étape de reconstruction peut être nécessaire avant leur étude. On peut citer d'anciennes sculptures ou bas reliefs, des Ostraca, et bien sûr de nombreux types de documents contenant de l'écriture ou des représentations picturales. Tous ces artéfacts anciens, s'ils sont retrouvés sous la forme de fragments, présentent souvent des caractéristiques communes:

1. Ils ont été dégradés par le temps et leurs conditions de conservation.
2. Des parties de l'artéfact originel peuvent être manquantes.
3. Ils contiennent de l'écriture ou d'autres représentations d'information.
4. Les contours des fragments résultant de leurs cassures ou déchirements ne forment pas des contours nets.

Dans cette thèse, nous restreignons notre étude aux documents fragmentaires. Nous proposons cependant une méthodologie qui s'abstrait du type de support, qui peut en principe être appliquée à tous types de supports fragmentaires. Notre méthodologie prend aussi en compte la possibilité d'adapter certains traitements aux spécificités des données si nécessaire.

Un des domaines de recherche englobant la plupart des travaux connexes à nos problématiques est celui de la *reconstruction de puzzles*. On y retrouve des sous-domaines appliqués à la reconstruction d'images fragmentaires, comme Le et Li (2019), qui propose un réseau de neurones combinant convolutions et blocs résiduels pour calculer la compatibilité entre des pièces de fragments d'images. Ils effectuent ensuite le recollage précis en utilisant des graphes et différentes techniques de fermeture de boucles. On peut aussi citer Paumard et al. (2018), qui propose une architecture de réseaux de neurones convolutive siamoise pour calculer les positions relatives de sous-images provenant d'œuvres d'art. Les caractéristiques des deux sous-images étudiées sont extraites puis combinées grâce à un *combination layer*. Cette combinaison est ensuite donnée en entrée à trois couches entièrement connectées, la dernière possédant 8 neurones et la fonction d'activation *Softmax*. On obtient donc à la fin, 8 valeurs qui correspondent à la probabilité de positions relatives des deux sous-images.

### 3.1.1 Pour les documents modernes

Concernant les documents, on retrouve principalement deux sous-domaines applicatifs: les documents déchirés manuellement ou mécaniquement à l'aide de machines (cf. Fig. 3.2). Dans le cas des documents déchirés manuellement, des travaux utilisent la forme de la déchirure ainsi que les caractéristiques locales présentes dans le voisinage des contours pour estimer des compatibilités entre fragments, comme

FIGURE 3.2 – Sur la gauche, un exemple de document déchiré mécaniquement, et sur la droite, manuellement. Source: stocklib.fr

Biswas et al. (2005) ou Richter et al. (2014). D'autres travaux exploitent et combinent différentes caractéristiques des documents, telles que la forme, la texture, la couleur, les lignes ou le type de papier (Diem et al. (2010), De Smet (2008)).

Une grande quantité de travaux portant sur les documents déchirés mécaniquement sont disponibles dans la littérature. On y retrouve des approches utilisant aussi des correspondances de caractéristiques locales au niveau des déchirures, comme Deever et Gallagher (2012) et Marques et Freitas (2009). Mais plus récemment, des approches exploitant la puissance de l'apprentissage profond sont apparues. Les auteurs de Paixão et al. (2020) proposent d'effectuer des évaluations de compatibilité entre bandes de papier déchirées mécaniquement en utilisant des méthodes de *Deep Metric Learning*, de manière *auto-supervisée*. Ils se concentrent sur les bordures des bandes, et construisent des paires positives et négatives de régions entre bandes, comme on peut le voir sur la Figure. 3.3 à gauche. Ces paires servent ensuite à entraîner des réseaux siamois. Ils fabriquent aussi leurs jeux de données d'entraînement automatiquement, ce qui en fait une approche *auto-supervisée*.

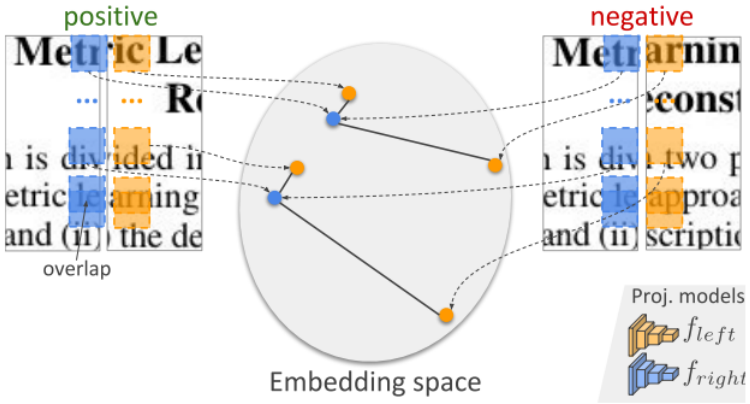



FIGURE 3.3 – Construction de paires de régions "compatibles" ou non à la bordure de bandes de papier déchirées mécaniquement. Source de l'image : Paixao et al. (2020)

Les méthodes utilisant l'apprentissage profond proposent, la plupart du temps, d'analyser des fragments issus d'un seul document afin de le reconstruire. C'est le cas des documents déchirés mécaniquement. Les fragments y sont souvent tous mélangés, car les documents déchirés sont jetés à la poubelle. Il est donc très important d'être capable de les trier avant de les recoller. Cette étape du problème est encore très difficile à résoudre. Les travaux se concentrent souvent sur des cas de figure plus simples pour éprouver leurs méthodes. C'est le constat que font les auteurs de Paixão et al. (2020) et c'est pour cette raison qu'ils proposent une méthode capable de fonctionner avec les fragments de plusieurs documents déchirés.

### 3.1.2 Pour les documents historiques

Les artéfacts historiques étudiés peuvent prendre la forme de documents sur papier, papyrus, ostraca, de peintures murales ou encore de sculptures (cf. Fig. 3.4). Les méthodes de reconstruction proposées dans la littérature exploitent aussi souvent les contours des fragments, comme Papaodysseus et al. (2002) et da Gama Leitao et Stolfi (2002). Ces méthodes doivent en plus traiter avec l'état dégradé de fragments anciens issus de fouilles archéologiques.

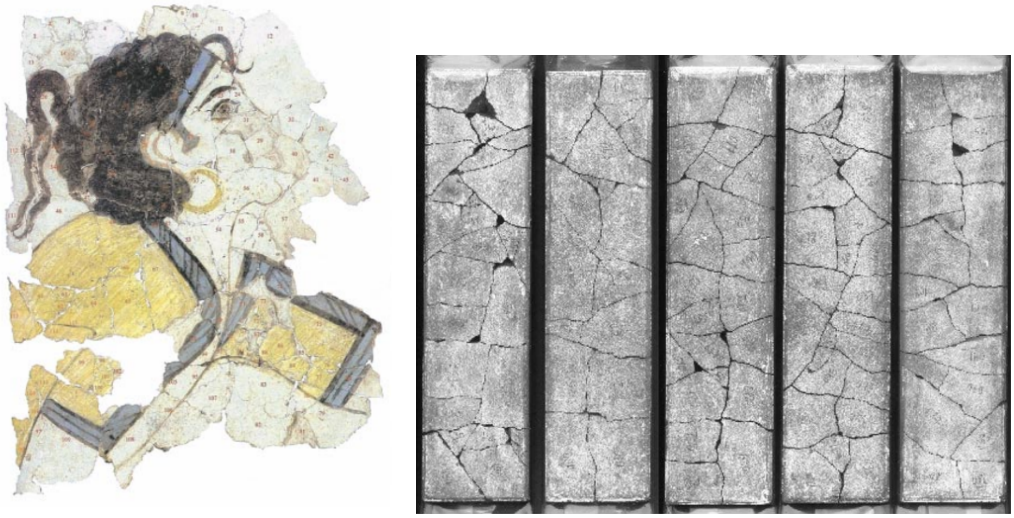


FIGURE 3.4 – À gauche, un exemple de peinture murale reconstruite (Source: Papaodysseus et al. (2002)). À droite, des fragments de poterie reconstruits (Source: da Gama Leitao et Stolfi (2002))

Lorsqu'on est confronté au problème de plusieurs documents fragmentaires qui sont mélangés entre eux, comme c'est souvent le cas à l'issue de fouilles archéologiques, une première étape de tri peut être nécessaire. Une fois qu'on a rassemblé les fragments appartenant aux mêmes documents, on peut le reconstruire par recollement. **Dans cette thèse, nous nous intéressons spécifiquement à la pre-**

**mière étape de tri.** Elle constitue une classification des fragments, mais dont on ne connaît pas les classes, ni leur nombre à l'avance.

Il est à noter que certaines approches cherchent à trier et à recoller géométriquement les fragments directement. C'est par exemple le cas de Abitbol et al. (2021). Les auteurs se servent de leurs mesures de compatibilité locale aux bordures des fragments pour sélectionner les fragments les plus pertinents pour le recollement. Ils font état de performances dégradées lorsque des écarts importants entre les fragments existent. Leur méthode obtient cependant de très bons résultats sur des fragments de papyrus, support qui est particulièrement sujet à des fragments manquants ou extrêmement dégradés.

Nous avons fait le choix de nous concentrer sur l'étape de tri, car il est rare que les papyrus (notre cadre applicatif final) soient constitués d'énormément de fragments (cf. Fig. 3.13 par exemple). Une fois qu'on a rassemblé les fragments constituant un document, il est relativement rapide pour une experte de reconstituer le puzzle avec un nombre limité de pièces.

Concrètement, un problème de recherche qui correspond à ce que nous souhaitons faire appartient au domaine du Content Based Image Retrieval (CBIR). Il est question de rechercher des images dans des bases de données en fonction de différentes modalités. On peut vouloir rechercher une image d'après une description textuelle (ex. Google Image), par une autre image (ex. recherche inversée Google), à partir d'un croquis, de concepts, etc (Alkhawlani et al. (2015)).

Nous cherchons les fragments pertinents dans une base de données d'après un fragment requête, similairement au domaine de la reconnaissance de visages (cf. 2.4.1). Pour rappel, l'idée est de représenter chaque image de l'ensemble de données sous la forme de vecteurs de caractéristiques comparables via une métrique de distance. Lors d'une nouvelle requête, on calcule le vecteur de caractéristiques sur l'image requête et on recherche les images ayant les vecteurs les plus proches dans la base de données (Saritha et al. (2019)). Aujourd'hui, ce domaine s'est en grande partie tourné vers des approches d'apprentissage automatique, et notamment l'apprentissage profond de métriques (cf. 2.4.1).

C'est une des raisons pour lesquelles nous avons choisi d'explorer l'approche de l'apprentissage profond de métriques (*DML*). À notre connaissance, les méthodes *DML* ont très peu été appliquées dans le contexte de la reconstruction de documents historiques. Nous en faisions le constat dans la section. 1.3.1. Dans les prochaines sections, nous décrivons la méthode que nous proposons pour calculer des suggestions d'appairages de fragments.

## 3.2 Suggestions d'appairages de fragments de documents par *DML*

### 3.2.1 Architecture siamoise pour l'apprentissage de distances entre patchs de documents

Nos réseaux siamois sont constitués de deux branches contenant chacune des enchaînements de couches de convolution/pooling. Les sorties de ces modules sont "aplaties" pour obtenir un vecteur de caractéristiques à une dimension qui est une projection de l'image d'entrée de la branche dans un espace latent. On calcule ensuite la valeur absolue de la différence élément à élément entre les deux vecteurs. Ce dernier vecteur est donné en entrée à deux couches denses dont la sortie finale est passée dans la fonction *sigmoid* pour obtenir une valeur entre 0 et 1. Cette valeur correspond à la probabilité que les deux images soient "similaires" d'après le modèle. La Figure. 3.5 résume ce descriptif.

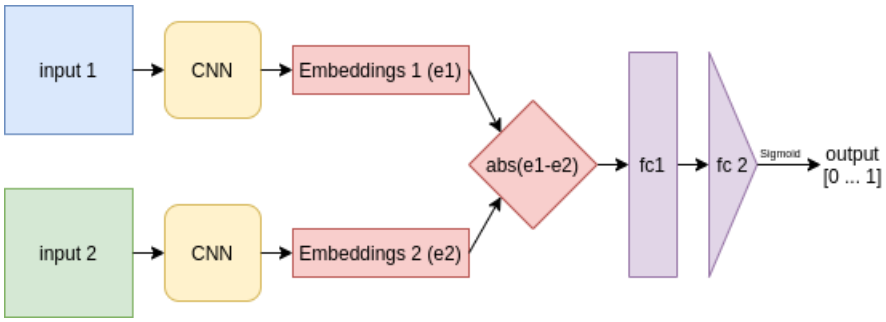



FIGURE 3.5 – Architecture globale de nos réseaux siamois permettant de calculer des distances entre patchs de documents.

Notre architecture diffère des réseaux siamois plus conventionnels par l'utilisation de ces deux couches entièrement connectées *fc1* et *fc2* sur la Figure 3.5. Comme vu dans le chapitre précédent, les réseaux siamois sont classiquement entraînés avec des fonctions de coûts de type *contrastive* ou *triplet*. Elles prennent directement en entrée les vecteurs de caractéristiques produits par les branches du réseau et guident l'entraînement de manière à converger vers des projections des entrées dans l'espace latent qui minimisent la distance entre deux entrées similaires et maximisent la distance entre deux entrées dissimilaires. Nous faisons le choix de combiner, en calculant la valeur absolue de la différence éléments à éléments, les deux vecteurs de caractéristiques en un vecteur qui est donné en entrée à deux couches entièrement connectées (fc1 et fc2 sur la Figure. 3.5). La toute dernière couche est réduite à un seul neurone qui passe par une fonction d'activation *Sigmoïde* de manière à obtenir en sortie un réel compris entre 0 et 1, qui correspond à la distance entre les deux entrées. Cela a l'avantage de produire des sorties normalisées, ce qui rend possible l'application d'un seuil pour classifier, comparé à des distances "arbitraires" que l'on

ne peut que comparer entre elles. Cela nous permet aussi d'entraîner nos modèles avec une fonction de coût conventionnellement utilisée pour la classification : la *binary cross-entropy loss*.

Nous avons choisi de fusionner les deux vecteurs de caractéristiques en calculant la valeur absolue de leurs différences élément à élément. Il aurait aussi été possible de concaténer les deux vecteurs pour les donner en entrée à la première couche entièrement connectée, mais nos tests ont mis en évidence l'apparition d'une asymétrie non désirée dans ce cas. On pouvait en effet obtenir des résultats différents en inversant deux entrées, ce qui va à l'encontre de la propriété de symétrie des métriques de distance énoncée dans la section 2.4.

### 3.2.2 Extraction de patchs de fragments

Nous utilisons une approche basée sur des patchs pour entraîner nos modèles. On cherche à représenter un fragment comme une collection de patchs les plus représentatifs et informatifs possibles. Une telle approche a deux principaux avantages : standardiser le format d'entrée des données dans le réseau sans déformer ou rogner les images et permettre un contrôle des propriétés des patchs qui sont extraits. On notera que les travaux de Bondi et al. (2017) suggèrent qu'avec un nombre suffisant de patchs bien choisis, une approche basée patchs peut atteindre des performances similaires à une approche utilisant les images entières. Cela tout en réduisant de manière significative la quantité de caractéristiques extraites par image, facilitant le processus d'optimisation et permettant l'usage de réseaux de neurones plus petits, ce qui entraîne au passage un gain de temps et de mémoire.

De par la nature irrégulière des déchirures présentes sur les documents avec lesquels nous travaillons, nous ne pouvons pas exploiter leurs formes pour faire des assemblages. De plus, des tests préliminaires ont montré que nos modèles entraînés avec des patchs présentant des contours de fragments ont tendance à considérer comme similaires deux patchs contenant des contours dont la forme est similaire. Ce n'est pas une modalité pertinente pour les raisons exposées ci-dessus, et même si cela l'était, il faudrait que les modèles apprennent plutôt à associer le complémentaire du contour plutôt qu'un contour ressemblant. Nous voulons donc extraire des patchs dans lesquels les contours des fragments ne sont pas présents. Il est de plus très important que les patchs extraits ne se chevauchent pas, car dans la réalité, les fragments d'un document déchiré ne comportent pas de recouvrement, cela viendrait à introduire des caractéristiques non réalistes dans le jeu de données.

Une implémentation de ces contraintes est donnée dans la Section. 3.4 pour sélectionner des patchs contenant du texte et ne contenant pas de contours.

### 3.2.3 Stratégie d'équilibrage des batchs

Si on construit toutes les paires de fragments possibles, on obtient beaucoup plus de paires *dissimilaires* que de paires *similaires*. Par exemple, une base de données contient 100 papyrus, ceux-ci sont en moyenne constitués de 3 fragments, pour un total d'environ 300 fragments. Pour un fragment donné, on peut donc construire 2 paires *similaires* (on ne crée pas de paires constituées du même fragment) et 297 paires *dissimilaires*. On a finalement $2 \times 300 = 600$ paires *similaires* et $297 \times 300 = 89100$ paires *dissimilaires*. Sur un total de 89700, les paires *similaires* ne représentent que 0.6% des paires formées. On obtient les mêmes ordres de grandeur lorsqu'on travaille à l'échelle des patchs.

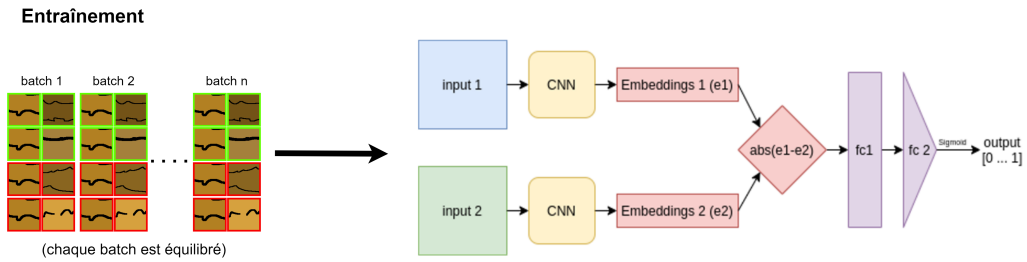



FIGURE 3.6 – Des batchs équilibrés sont construits. Ils contiennent chacun autant de paires similaires (en vert) que de paires dissimilaires (en rouge). Ces batchs sont utilisés pour l'entraînement de nos modèles siamois.

Lors de la phase d'entraînement, on donne au réseau des paires de patchs *similaires* et *dissimilaires*. On se rend facilement compte que si on se contente de remplir les *batchs* de paires choisies aléatoirement, on ne tirera que très rarement une paire *similaire*. C'est un problème, car en ne voyant quasi exclusivement que des paires *dissimilaires*, l'algorithme d'optimisation va rapidement faire converger le modèle vers une sortie toujours négative. En effet, en produisant une prédiction négative systématiquement, la fonction de coût sera toujours très faible, car les rares paires *similaires* seront les seules occurrences de mauvaises prédictions et seront noyées dans une masse de bonnes prédictions sur les paires *dissimilaires*.Un modèle n'apprend donc essentiellement rien dans cette configuration. On peut résoudre ce problème de deux manières : équilibrer les batchs ou pondérer la fonction de coût (cf. Fig. 3.6). Dans le premier cas, on va sur-échantillonner la classe minoritaire (paires *similaires*) ou sous-échantillonner la classe majoritaire (paires *dissimilaires*), ou les deux, afin de contrôler l'équilibre des deux classes au sein des batchs. Dans le deuxième cas, on assigne plus d'importance aux prédictions sur la classe minoritaire lors du calcul de la fonction de coût en mettant un poids plus élevé à ces dernières.

Ces deux stratégies forcent le modèle à apprendre des caractéristiques utiles pour distinguer les deux classes. Il faut cependant garder à l'esprit qu'on apprend ici sur

une distribution de données biaisée, qui n'est pas représentative de la distribution réelle. C'est pour cela que l'on n'utilise pas les mêmes métriques d'évaluation à l'entraînement qu'à l'évaluation. Des détails sur les procédures d'évaluation sont donnés dans la section 3.2.5.

Après avoir mené des expériences, la stratégie d'équilibrage est la seule qui fonctionne pour nous. Nous n'avons pas pu entraîner convenablement nos modèles en pondérant la fonction de coût. Nous nous concentrons donc sur l'équilibrage des batchs dans la suite.

### 3.2.4 Utilisation de distances entre patchs pour calculer une distance entre fragments

**Note** : On parlera ici de résultats *niveau-patch* (*patch-level*) pour désigner les scores bruts renvoyés par le classifieur entre deux patchs, et de résultats *niveau-fragment* (*fragment-level*) pour désigner des scores de correspondance entre fragments calculés à partir des scores *niveau-patch*.

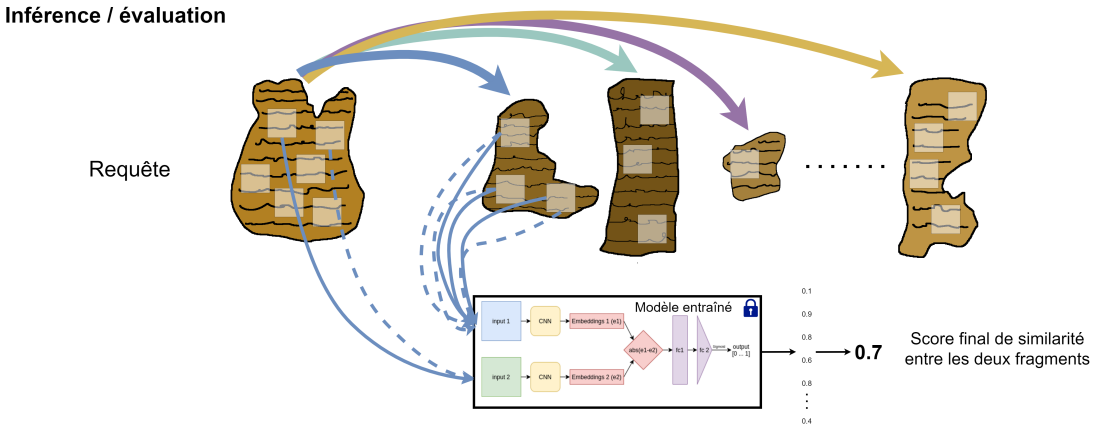



FIGURE 3.7 – Procédure d'évaluation des résultats. Étant donné un fragment requête, on le compare à tous les autres fragments du jeu de données. On calcule un score de similarité entre deux fragments à partir de la moyenne des scores des paires de patchs, calculée grâce au modèle siamois déjà entraîné.

Comme nous venons de l'expliquer, notre réseau siamois travaille sur des paires de *patchs*. Il évalue la similarité des deux patchs qui lui sont donnés en entrée. Seulement, on cherche ultimement à déterminer si deux fragments proviennent du même document, c'est-à-dire des correspondances *niveau-fragment*. Lors de la phase d'évaluation, on a seulement une collection de fragments et on souhaite obtenir des scores de similarité de tous les fragments entre eux. On considère donc chaque fragment comme une requête, et on le compare à tous les autres de la collection (cf. Fig. 3.7).

Pour obtenir un score de similarité entre deux fragments, on extrait un certain nombre de patchs de chacun de ces fragments et on construit toutes les paires de patchs possibles entre les deux fragments (sans construire de paires de patchs provenant du même fragment). On donne toutes ces paires en entrée au réseau siamois déjà entraîné, qui va produire des scores de similarité pour chacune des paires. Ensuite, on souhaite traiter cette liste de scores afin d'en extraire un unique score de similarité entre deux fragments. L'approche la plus basique consiste à calculer la moyenne des scores comme décrit sur la Figure. 3.8. Tout d'abord sont calculées les distances entre le patch 1 et les 3 patchs du fragment B puis la distance entre le patch 2 et les 3 patchs du fragment B. En faisant la moyenne de ces 6 distances, on obtient une distance entre le fragment A et le fragment B.

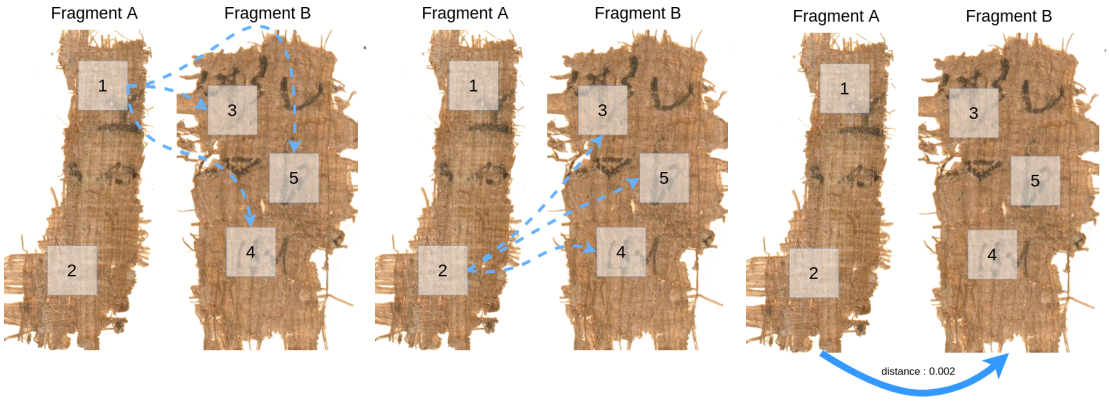



FIGURE 3.8 – Calcul de la distance entre deux fragments. On calcule les distances de toutes les paires de patchs possibles entre les deux fragments. La moyenne de toutes ces distances est la distance entre les deux fragments.

### 3.2.5 Métriques pour évaluer notre méthode de calcul entre fragments de documents

Nous pouvons maintenant calculer des distances entre fragments. À présent, afin de correctement évaluer les performances de notre approche, il est critique de choisir un protocole d'évaluation et des métriques cohérentes avec la tâche que l'on évalue et les propriétés des données. Comme décrit dans la section 3.2.3, notre problème est intrinsèquement déséquilibré, on peut former beaucoup plus de paires *dissimilaires* que de paires *similaires*. On équilibre donc les batchs lors de la phase d'entraînement afin de permettre aux modèles de converger. On se sert alors de la métrique classique qu'est la précision pour surveiller le processus d'apprentissage. Puisqu'on est dans un cas équilibré, cette métrique est pertinente.

Cependant, lors de l'évaluation des performances en configuration réelle, c'est-à-dire évaluer si l'on mesure correctement une distance entre fragments, on évalue toutes les paires de fragments possibles. Utiliser la précision devient difficile à inter-

préter à cause du grand déséquilibre de classes. C'est aussi le cas de l'Area Under Curve (AUC) de la Reciever Operating Characteristic (ROC).

### Précision

La précision, ou *accuracy* en anglais, est une métrique classique très utilisée pour évaluer la performance d'un classifieur en *Machine Learning*. Elle correspond au nombre de prédictions correctes divisé par le nombre de prédictions faites :

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions\ to\ be\ made} = \frac{TP + TN}{TP + FP + TN + FN}$$

### Mean Average Precision (mAP)

La mAP mesure la capacité d'un algorithme d'*Information Retrieval* à retrouver *rapidement* (parmi les mieux placés dans la liste classée de prédictions) les éléments pertinents, à savoir les Vrais Positifs. Elle est très sensible à la présence de Faux Positifs tôt dans la liste classée de prédictions et pénalise un algorithme qui classe un Faux Positif avec un score élevé. La Mean Average Precision (mAP) diminue lorsque des Vrais Positifs sont classés après des Faux Positifs.

On compare chaque fragment avec tous les autres fragments du jeu de test. La liste triée dans l'ordre décroissant des scores correspondant à un fragment comparé à tous les autres est appelée une requête. Pour chaque requête, on calcule la Average Precision (AP) :

$$AP = \frac{1}{R} \sum_{r=1}^{S} Pr(r) \times rel(r)$$

Avec :
- $R$ : Le nombre de fragments pertinents (*relevant*)
- $S$ : La taille de la requête
- $Pr(r)$ : La précision au rang $r$ (définie dans 3.2.5 : **Precision-Recall**)
- $rel(r)$ : Une fonction retournant 1 si le fragment du rang $r$ est pertinent, 0 sinon

On calcule ensuite la moyenne de ces scores, ce qui donne la mAP. La AP est une approximation de la surface sous la courbe de precision-recall, ainsi la mAP correspond à une moyenne des surfaces sous la courbe de precision-recall pour toutes les requêtes.

### Top-1 accuracy

Cette métrique indique simplement si en moyenne, l'élément classé en premier dans la requête est pertinent. Pour chaque requête, le score est 1 si l'élément est

pertinent, 0 sinon. On fait ensuite la moyenne des scores pour obtenir la top-1 accuracy.

#### Precision at k (Pr@k)

Cette métrique indique la proportion d'éléments pertinents retrouvés pour une requête de taille $k$. Pour chaque requête triée, on calcule :

$$Pr@k = \frac{R_k}{min(R_N, k)}$$

Avec $R_k$ le nombre d'éléments pertinents jusqu'au rang $k$ et $N$ le nombre total d'éléments dans la requête. On calcule ensuite la moyenne de ce score pour chaque requête. Au niveau du dénominateur, l'intérêt d'utiliser $min(R_N, k)$ est d'obtenir un score parfait (de 1) s'il y a moins d'éléments pertinents au total que d'éléments dans la requête et qu'ils ont tous été retrouvés.

## 3.3 Création d'une base d'apprentissage

Les expériences présentées par la suite sont faites sur deux bases de données aux caractéristiques différentes. Nous avons construit une base de données d'entraînement à partir de la collection de papyrus de l'Université du Michigan. Cela a nécessité un travail de tri et de pré-traitement qui est décrit dans la section 3.3.1. Nous avons aussi utilisé le jeu de données fourni par la compétition *HisFrag*, qui est constituée de fragments de documents anciens qui ne sont pas des papyrus. Ce jeu de données pouvant être utilisé tel quel, nous nous contenterons d'en donner une description dans la section 3.3.2.

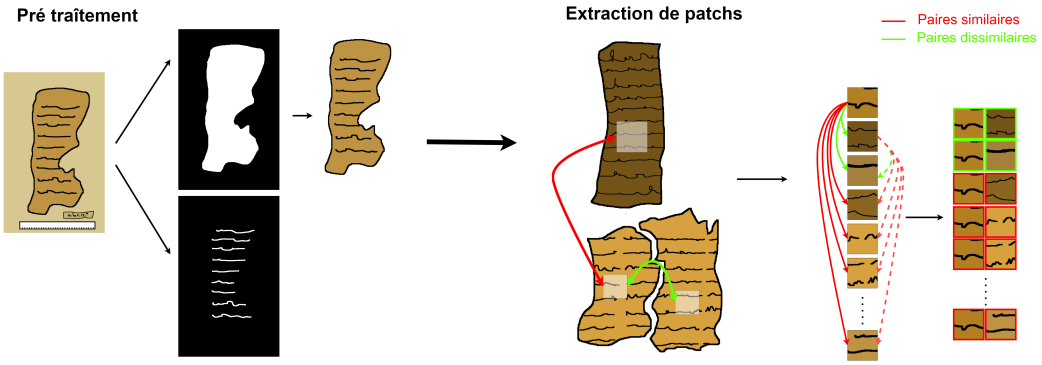



FIGURE 3.9 – À partir de l'image originale, un masque du fragment ainsi qu'un masque des sous-lignes sont extraits. Ces masques permettent de sélectionner des patchs dans les fragments dans lesquels on peut contrôler la présence de texte et de bordure.

L'objectif premier de ces pré-traitements est d'extraire des patchs de fragments de manière contrôlée afin de construire des paires de patchs qui serviront à l'entraînement des réseaux siamois (cf. Fig. 3.9).

### 3.3.1 Création d'une base de données d'entraînement issue des humanités numériques

L'Université du Michigan met à disposition sa collection de papyrus numérisés [2] sous Licence Creative Commons 3.0 et autorise l'utilisation de ses images dans le cadre de travaux éducatifs ou de recherche. C'est la plus grande collection de papyrus d'amérique du Nord, avec plus de 18.000 fragments anciens (cf. Tableau. 3.1) écrits entre 1000 AEC et 1000 EC, principalement écrits en 9 langues (Arabe, Démotique, Grec, Latin, Copte, Hiératique, Hébreu, Copte ancien et Copte du Fayoum). Ces documents proviennent de fouilles effectuées entre les années 1920 et 1940 en Égypte et contiennent entre autre la plus vieille copie connue des Épîtres de Paul. Le projet de numérisation de la collection a commencé en 1991 initialement en collaboration avec les universités de Columbia, Duke et Yale. La quantité de données disponibles est très différente en fonction de la langue (cf. Tableau. 3.1). Nous avons donc choisi de ne conserver que les papyrus écrit en *grec*, car c'est de loin la langue la plus représentée ($\approx 87\%$ des fragments sont écrits en *grec*).

| Langues | Nombre d'images |
|---|---|
| Greek | 14890 |
| Coptic | 1140 |
| Arabic | 404 |
| Demotic | 163 |
| Latin | 138 |
| Fayumic_Coptic | 129 |
| Demotic.Greek | 63 |
| Coptic.Greek | 26 |
| Greek.Latin | 20 |
| Hieratic | 15 |
| old_Coptic | 12 |
| Arabic.Greek | 10 |
| Autres | 19 |
| **Total** | 17029 |

Tableau 3.1 – Nombre d'images par langues dans la collection de l'Université du Michigan. Lorsque plusieurs langages sont séparés par un point, cela signifie que le document contient plusieurs langues

2. https://www.lib.umich.edu/locations-and-hours/papyrology-collection (dernier accès : 15/09/2021)

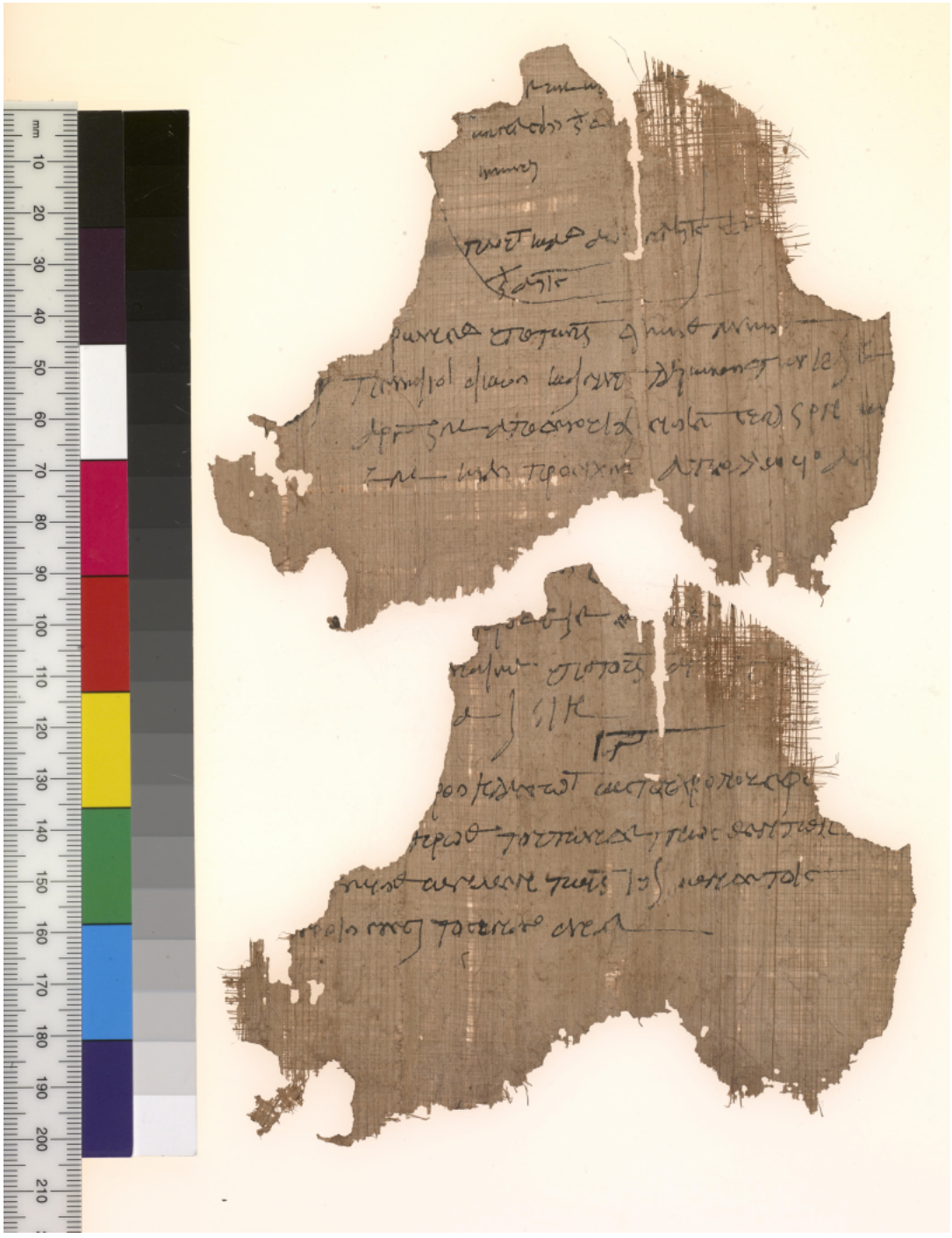


FIGURE 3.10 – Exemple d'image brute issue de la base de données *Michigan*.

Le site web de la collection de papyrus de l'Université du Michigan ne propose pas de télécharger directement l'entièreté de la base de données, sous la forme d'une archive par exemple. Cependant, un système de recherche par mots clés de papyrus

est disponible [3], ce qui nous a permis d'exploiter la structure des requêtes web pour récupérer la base de données automatiquement à l'aide d'un script.

Les images récupérées contiennent une règle graduée, une échelle de couleurs et une échelle de niveaux de gris (cf. Fig. 3.10). Ces images étant des photos qui ont été faites avec des appareils différents, à différentes époques et avec des niveaux de zoom différents, un traitement est nécessaire pour connaître la taille réelle des fragments (en centimètres) afin de mettre toutes les images à la même densité de pixels. En uniformisant les images de la sorte, on limite un potentiel biais lors de l'apprentissage lié à la taille des caractères manuscrits. En effet, les photographies ne sont pas prises avec la même distance entre le papyrus et l'appareil photo. En fonction de la taille du fragment photographié, l'appareil est plus ou moins proche alors que la définition de l'image ne change pas. Lorsque deux fragments de dimensions différentes, mais appartenant au même papyrus sont photographies séparément, l'écriture apparaît de taille différente alors qu'en réalité elle est de même taille. On souhaite donc que toutes les images soient à la même échelle en termes de pixels par centimètres.

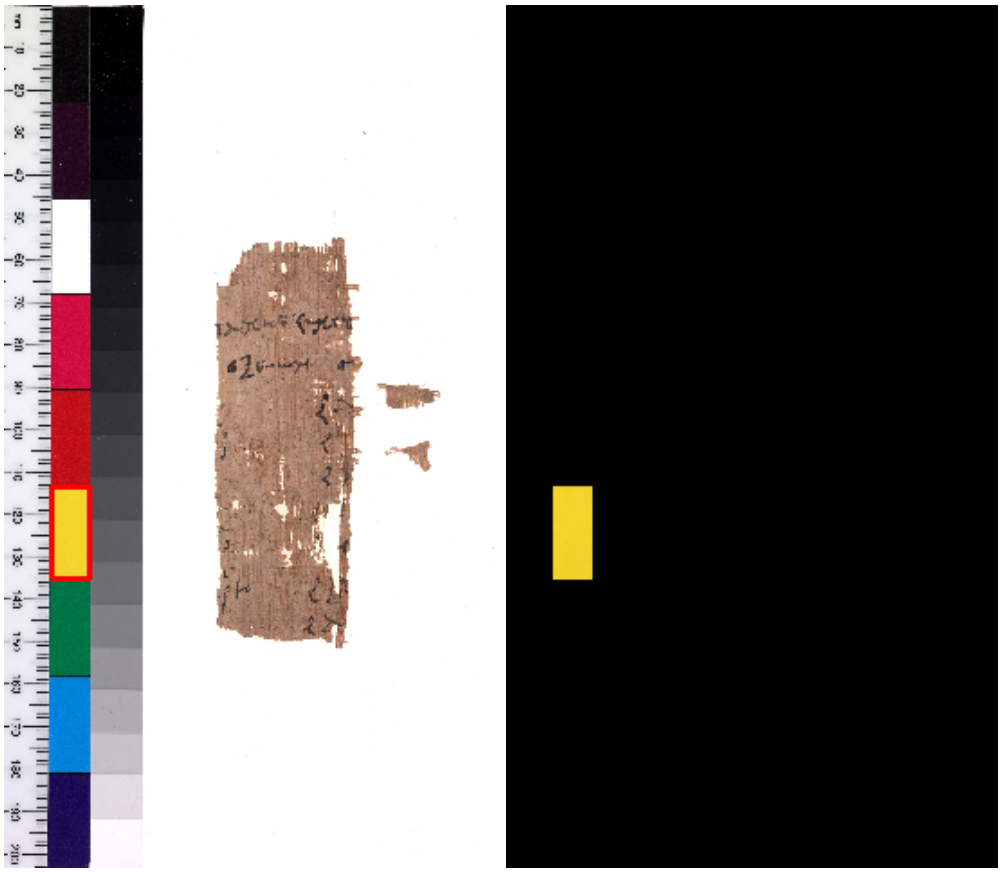


FIGURE 3.11 – Extraction d'un rectangle de couleurs de hauteur réelle connue (23 mm) pour déterminer la densité de pixels (en pixels par centimètres) de l'image.

3. https://quod.lib.umich.edu/a/apis

Nous déterminons la taille réelle des fragments en centimètres en analysant les cases de l'échelle de couleurs. On observe que la hauteur d'une de ces cases rectangulaires est de 23mm (cf. Fig. 3.11) grâce à la règle graduée. Nous partons du principe que la règle et l'échelle de couleurs ont la même taille dans toutes les images de la base de données. On utilise une simple heuristique de correspondance de couleurs pour isoler une couleur choisie dans l'échelle de couleur. En connaissant la hauteur d'un rectangle de couleur en pixels et en centimètres, on détermine la densité en pixels par centimètres. On se sert de cette information pour redimensionner toutes les images de la base afin qu'elles aient toutes la même densité de pixels.

On extrait ensuite les fragments des images. Comme on peut le voir sur la Figure. 3.10, certaines images contiennent plusieurs fragments, qu'on sépare donc en autant d'images que de fragments (cf. Fig. 3.12). Une fois l'extraction des fragments terminée, on obtient 6607 images contenant un fragment unique, qui composent 3823 papyrus.

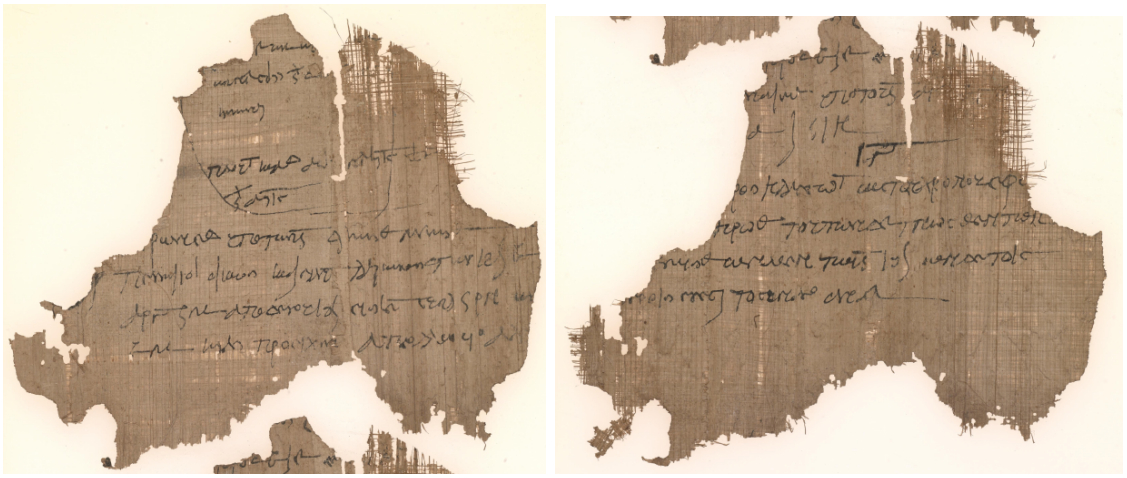


FIGURE 3.12 – Deux fragments de papyrus séparés en deux images distinctes

### 3.3.2 Le jeu de données de la compétition *Hisfrag*

Afin de ne pas limiter nos expériences aux papyrus et d'élargir notre proposition, nous avons choisi d'utiliser une autre base de données en plus de celle du *Michigan*. Publiée à l'occasion de la conférence *ICFHR2020* pour la compétition "*Competition on Image Retrieval for Historical Handwritten Fragments*" (Seuret et al. (2020)), la base de données, que nous désignerons par le nom de la compétition, *HisFrag*, est constituée de plus de 120.000 fragments de documents. Ces fragments ont été créés de façon synthétique en découpant automatiquement des documents entiers à l'aide d'un algorithme basé sur le "Diamond square algorithm" [4]. On peut d'ailleurs

4. https://github.com/seuretm/diamond-square-fragmentation

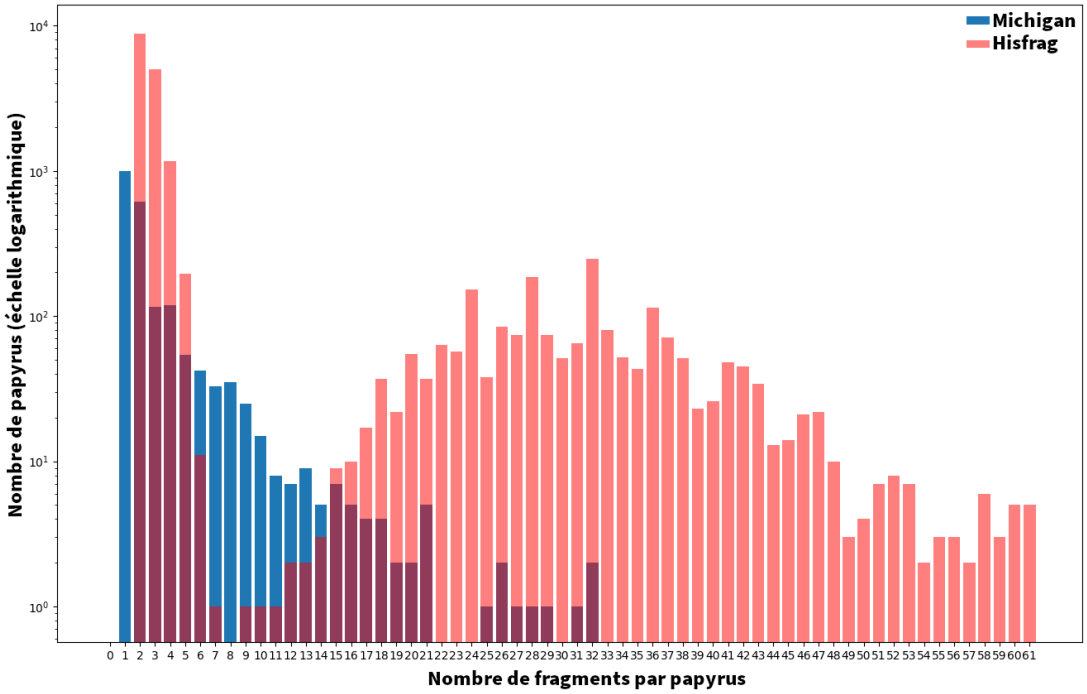



FIGURE 3.13 – Distribution du nombre de fragments par papyrus pour les bases *Michigan* et *Hisfrag*.

remarquer que la distribution du nombre de fragments par papyrus très différente de celle de la base *Michigan*, comme on peut le voir sur la Figure. 3.13. Cette compétition propose deux tâches, "*writer identification*" et "*page fragment retrieval*" à partir des mêmes données, les annotations étant différentes suivant la tâche choisie.

Classiquement, le jeu de test d'un jeu de données est constitué d'un sous-ensemble de la base de données, qui est mis de côté et qui ne sera jamais utilisé ni pour l'entraînement ni pour la validation. Les auteurs de la base *Hisfrag* ont décidé d'aller plus loin en constituant le jeu d'entraînement/validation et le jeu de test à partir de bases de données différentes.

Le jeu d'entraînement utilise des images provenant de *Historical-IR19* (Christlein et al. (2019)). Ces images sont des documents manuscrits, des chartes et des lettres de sources différentes. Le jeu de test est constitué de lettres fournies par la bibliothèque de l'université de Bâle[5], et de manuscrits provenant d'un corpus de livres datant du Moyen Âge. Les deux jeux de données ont des aspects très différents, comme on peut le constater sur la Figure. 3.14. Les auteurs peuvent produire deux types de fragments : des fragments de forme aléatoire et des fragments de forme rectangulaire. Il semble qu'ils aient choisi de produire beaucoup plus de fragments de forme aléatoire dans le jeu de test que dans le jeu d'entraînement.

5. https://www.unibas.ch/de

Finalement, le jeu d'entraînement contient 101.706 fragments générés à partir de 17222 documents et ayant été écrits par 8717 auteurs. Le jeu de test contient 20019 fragments générés à partir de 2732 documents et ayant été écrits par 1152 auteurs.

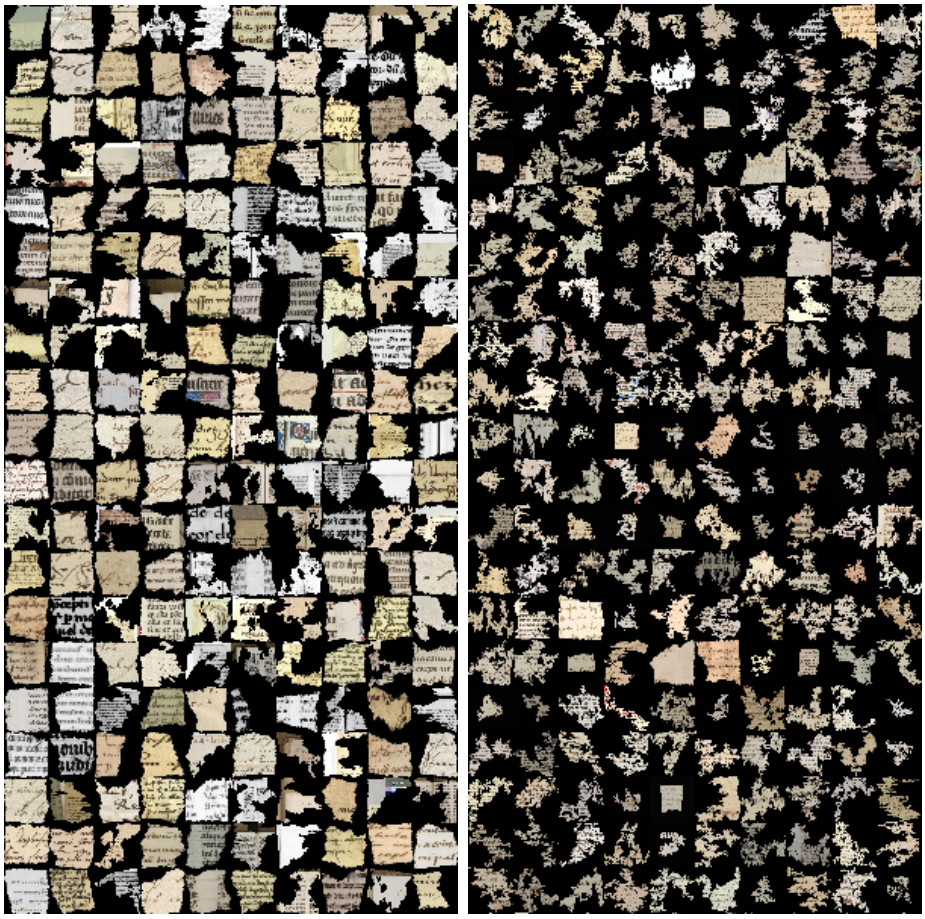


FIGURE 3.14 – Échantillonnage de fragments provenant du jeu d'entraînement (à gauche) et du jeu de test (à droite) de la base *Hisfrag*

### 3.3.3 Détecter des lignes avec ARU-Net

Si l'on souhaite contrôler la présence de texte dans les patchs, il nous faut savoir où se trouve le texte dans l'image. Dans un article de 2019, Grüning et al. (2019) proposent une méthode en deux étapes d'extraction de sous-lignes (des lignes virtuelles sous le texte) sur des documents manuscrits anciens. La première étape effectue de la segmentation de pixels, c'est-à-dire une classification binaire des pixels en deux classes : sous-ligne ou non. On obtient alors une image sur laquelle les pixels blancs sont classifiés comme appartenant aux sous-lignes, comme on peut le voir sur la Figure. 3.15. La seconde étape utilise des techniques de traitement d'image plus traditionnelles pour extraire de cette segmentation des sous-lignes "géométriques" précises. Nous n'avons utilisé que la première étape : **ARU-Net**.

**ARU-Net** est un réseau de neurones profond basé sur une architecture **U-Net** (Ronneberger et al. (2015)). Les auteurs proposent une évolution de cette architecture en y ajoutant des **blocs résiduels** et des **Attention Networks**.

FIGURE 3.15 – Exemple de détection des lignes de texte avec *ARU-Net*

Les **U-Net** sont des architectures qui ont d'abord été développées pour la segmentation d'images biomédicales, et ensuite reprises pour d'autres types de tâches, la plupart du temps de segmentation. Comme on peut le voir sur la Figure 3.16, la première partie de l'architecture est un enchaînement classique de couches de convolutions avec des opérations de *pooling* réduisant la dimension spatiale des cartes de caractéristiques. La deuxième partie est symétrique à la première, mais augmente progressivement la dimension pour arriver à une sortie finale de mêmes dimensions que l'entrée. On a ainsi un "chemin de contraction" qui extrait les caractéristiques importantes et un "chemin de dilatation" qui construit une sortie la plus proche possible de la vérité terrain à partir de ces caractéristiques.

Nous avons expliqué la nature des **blocs résiduels** dans la section 2.2.2. Les **Attention Networks** quant à eux sont des architectures de réseaux de neurones cherchant à imiter le mécanisme cognitif d'attention (Bahdanau et al. (2014)). L'idée étant d'apprendre à concentrer la "vigilance" du réseau sur des zones spécifiques des données d'entrée en construisant des "cartes d'attention". Dans le cadre d'images, la "carte d'attention" est blanche sur les zones importantes, et noire sur les zones peu importantes, on peut alors pondérer l'impact de la zone parcourue par une fenêtre de convolution dans le processus global d'apprentissage.

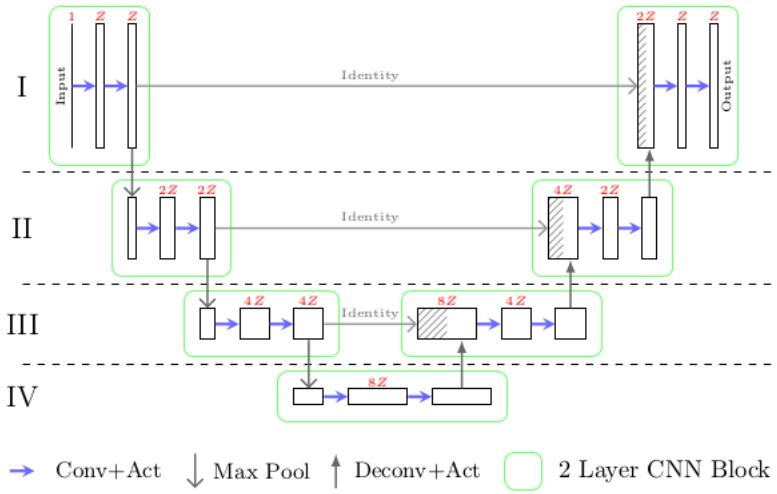



FIGURE 3.16 – Une architecture de U-Net. Source : Grüning et al. (2019)

Nous avons ré-entraîné **ARU-Net** [6] avec des images de papyrus. Les auteurs mettent l'accent sur le fait qu'une relativement petite quantité de données annotée est suffisante pour atteindre des performances acceptables. Nous avons donc annoté environ 200 images, c'est-à-dire que nous avons construit à la main des masques binaires sur lesquels un pixel est blanc s'il appartient à une "sous-ligne".

Après entraînement pendant 100 epochs sur une machine ayant un GPU Nvidia Titan X (Pascal) avec 12GB de VRAM, nous obtenons rapidement des résultats satisfaisants pour notre utilisation, comme on peut le voir sur la Figure. 3.17. Il ne nous est pas nécessaire d'aller plus loin, car nous cherchons seulement à avoir une idée approximative d'où se trouvent les lignes de texte, et ces performances sont suffisantes pour produire des patchs contenant du texte.

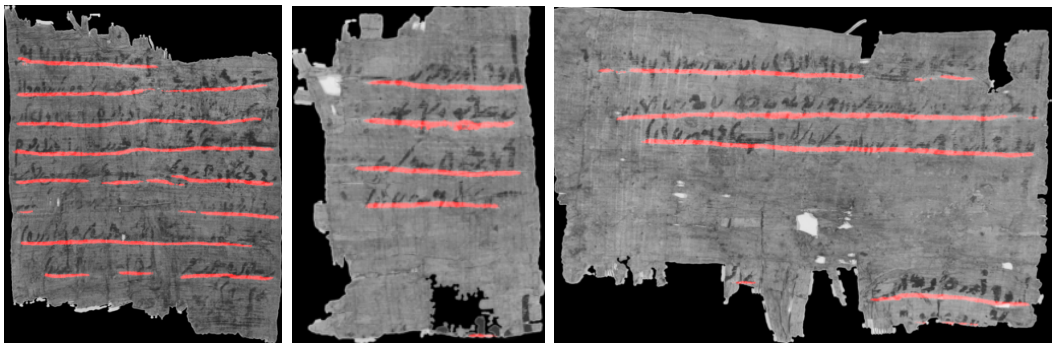


FIGURE 3.17 – Résultats d'extraction de sous-lignes avec ARU-Net

6. Disponible ici : https://github.com/TobiasGruening/ARU-Net

## 3.4 Apprendre des distances entre patchs

Nos travaux ont donné lieu à une première publication (Pirrone et al. (2019)) dans le *workshop HIP2019* (*International Workshop on Historical Document Imaging and Processing*). Dans cet article, nous cherchons à savoir s'il est possible d'utiliser une architecture de type siamois pour déterminer si deux patchs proviennent du même fragment. Nous souhaitons aussi évaluer l'impact de la présence de texte dans les patchs sur les performances du modèle.

### 3.4.1 Sélectionner des patchs contenant du texte et ne contenant pas de contours

On souhaite attribuer un score à chaque patch en fonction de la quantité de texte et de la quantité de "fond" qu'il présente. Pour cela, on utilise les masques de sous-lignes produits grâce à *ARU-Net* (cf. Section. 3.3.3), et des masques de segmentation approximatifs calculés à la volée, qui sont suffisants pour avoir une idée de la quantité de "fond" présent sur le patch. Pour chaque patch, on peut donc calculer la proportion de pixels appartenant au "fond" et celle appartenant aux sous-lignes. On calcule ainsi un score grâce à la formule de l'équation 3.1.

$$patchScore = a \times (\frac{nbTextPixels}{nbPixels}) + b \times (1 - \frac{nbFondPixels}{nbPixels}) \tag{3.1}$$

On peut pondérer l'importance de la présence de texte et de l'absence de fond dans les patchs grâce aux paramètres $a$ et $b$, qui sont par défaut à 1. On peut ainsi classer les patchs en fonction de ce score dans l'ordre décroissant, ceux ayant le meilleur score étant les "meilleurs" patchs d'après nos critères. Il ne reste plus qu'à prendre les $n$ premiers en fonction du nombre de patchs par fragments que l'on souhaite extraire (cf. Fig. 3.18).

FIGURE 3.18 – Exemples de patchs de taille $64 \times 64$ choisis en fonction du fait qu'ils contiennent de l'écriture, et ne sont pas positionnés sur un bord du fragment.

### 3.4.2 Validation de la méthode proposée : tests sur *Papy-S-Net*

Nous nous sommes inspirés du travail de *Koch et al.* pour construire une architecture de réseaux siamois que nous avons appelée *Papy-S-Net*. Dans Koch (2015),

les auteurs proposent un réseau siamois profond pour de la reconnaissance de caractères manuscrits. Plus précisément, ils cherchent à déterminer si deux images de caractères manuscrits correspondent au même caractère en utilisant la base de données *Omniglot* proposée par Lake et al. (2015).

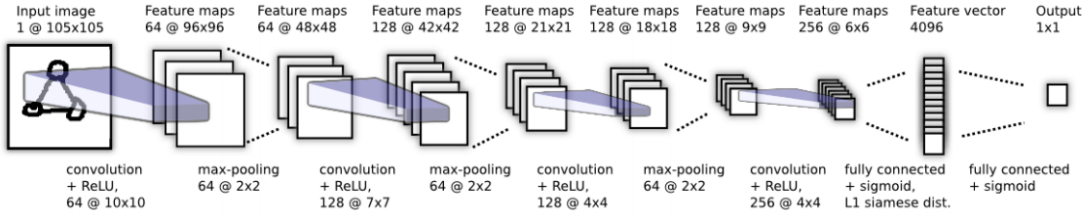



FIGURE 3.19 – Architecture proposée par *Koch et al.*[7]

Nous proposons une architecture de réseaux siamois dont les branches sont dérivées de l'architecture *VGG* (Liu et Deng (2015)) (cf. Fig. 3.20). Cette architecture reprend l'enchaînement de plusieurs couches de convolution avant *pooling* que l'on retrouve dans *VGG*. Chaque branche est constituée de trois doubles blocs de convolution/pooling, la définition des cartes de caractéristiques est divisée par 2 à chaque étage par *max pooling* (cf. Fig. 3.20). Nous avons choisi de comparer notre architecture avec celle proposée par *Koch et al.* (cf. Fig 3.19) afin d'avoir un premier élément de comparaison.

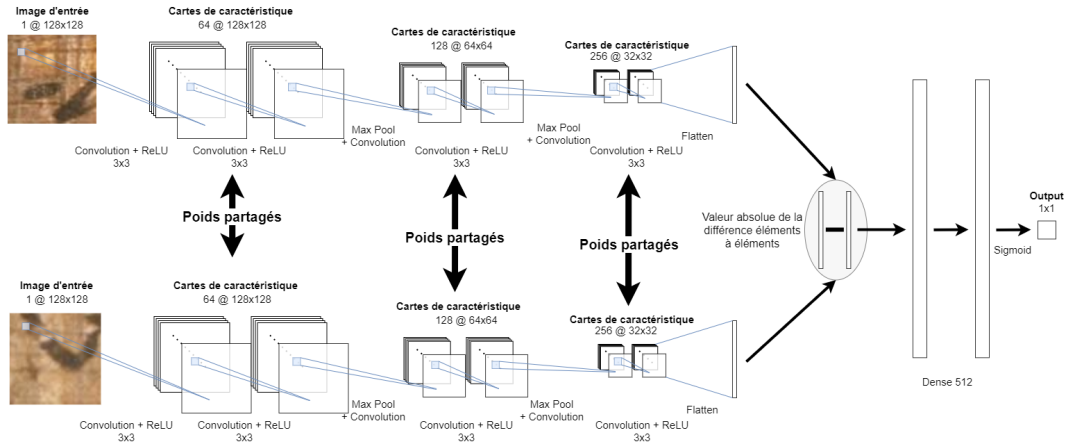



FIGURE 3.20 – Architecture de Papy-S-Net

**Note:** Nous parlons dans la suite d'information *niveau-fragment* et d'information *niveau-document*. Cela désigne le type de vérité terrain que l'on peut utiliser pour entraîner nos modèles. L'information *niveau-fragment* correspond à la connaissance

7. La figure a été directement reprise de l'article Koch (2015)

que deux patchs proviennent du même fragment, et l'information *niveau-document* correspond à la connaissance que deux fragments appartiennent au même document.

Dans un premier temps, une base de données de 500 fragments de papyrus provenant de la collection de l'Université du Michigan a été construite. Nous l'appellerons ici *B500*. Nous n'avons que de l'information *niveau-fragment* sur cette base, nos premiers tests ont pour but d'estimer s'il est possible d'utiliser des réseaux de type siamois afin de déterminer si deux patchs proviennent du même fragment. L'évaluation se fait donc au niveau des patchs sur la partie test de la base *B500*, on calcule les faux positifs, vrais positifs, faux négatifs et vrais négatifs directement sur les paires de patchs. C'est-à-dire qu'étant donnés deux patchs, s'ils proviennent du même fragment et que le score renvoyé par le modèle est au-dessus d'un seuil (ici 0.5), alors c'est un vrai positif. S'ils ne proviennent pas du même fragment et que le score renvoyé par le modèle est au-dessus du seuil, alors c'est un faux positif, etc.

Ici, nous nous intéressons aussi à l'impact de la présence ou non de texte dans les patchs extraits. Nous avons donc construit trois jeux de données, un contenant seulement des patchs contenant du texte, un contenant seulement des patchs ne présentant pas du tout de texte et un jeu aléatoire, tous les trois approximativement de même taille. À ce stade, nous n'imposons pas de contrainte sur le nombre de patchs que l'on extrait par fragments. Nous verrons plus tard qu'il est important de réguler cela, car suivant la taille des fragments, ce nombre peut énormément varier (entre 2 et 123), introduisant possiblement des biais statistiques en faveur des gros fragments.

Les patchs créés sont de taille $128 \times 128$ pixels. On extrait tous les patchs non chevauchants possibles de tous les fragments. On construit ensuite les jeux de données de manière classique avec trois sous jeux *Entraînement/Validation/Test*, avec environ 90% des fragments dans les jeux *Entraînement* et *Validation*. On construit ensuite autant de paires *similaires* que de paires *dissimilaires* sur les jeux *Entraînement* et *Validation* et toutes les paires possibles sur le jeu *Test*, pour les raisons exposées dans la Section. 3.2.3.

Les deux modèles [8] sont implémentés dans la version 1.13 du framework de *Deep Learning Tensorflow* [9] en Python 3 [10]. Les expériences sont menées sur les serveurs de calcul dédiés au *Deep Learning* du Laboratoire Bordelais de Recherche en Informatique (LaBRI), disposant de *Nvidia* [11] *Titan X Pascal* avec 12 Go de Video Random Access Memory (VRAM) chacune. Les entraînements sont réalisés avec une taille de *batch* de 64, un *learning-rate* de 0.0001 et en utilisant l'implémentation

8. Le code source des expériences peut être trouvé ici : https://github.com/apirrone/Self-Supervised-Deep-Metric-Learning-for-ancient-papyrus-fragments-retrieval
9. https://www.tensorflow.org
10. https://www.python.org
11. https://www.nvidia.com

*Tensorflow* de l'optimiseur *AdamOptimizer* (Kingma et Ba (2014)), et ce pendant 80 epochs.

| | Aléatoire | | Sans Texte | | Avec Texte | |
|---|---|---|---|---|---|---|
| | Papy-S-Net | Koch | Papy-S-Net | Koch | Papy-S-Net | Koch |
| Vrai Pos. | 0.80 | 0.74 | 0.75 | 0.76 | **0.82** | 0.72 |
| Vrai Neg. | 0.92 | 0.88 | 0.82 | 0.87 | **0.94** | 0.86 |
| Faux Pos. | 0.09 | 0.12 | 0.08 | 0.13 | **0.06** | 0.14 |
| Faux Neg. | 0.20 | 0.26 | 0.25 | 0.24 | **0.18** | 0.28 |

Tableau 3.2 – Comparaison des performances sur la base *B500* obtenues par Papy-S-Net et Koch et. al. En gras le meilleur résultat pour chaque métrique, et en souligné le pire.

Comme on peut le voir sur le tableau 3.2, les meilleures performances en termes de vrais positifs, vrais négatifs, faux positifs et faux négatifs sont obtenues en utilisant notre architecture, entraînée avec des patchs contenant tous du texte. Les plus mauvaises performances sont obtenues avec l'architecture de Koch et. al., aussi entraînée avec des patchs contenant tous du texte. Alors que les différences de performances entre ces deux exemples sont significatives (entre 8 et 10 points), les différences respectives entre les approches aléatoires et avec texte sont globalement plus faibles. En prenant en compte le coût de pré-traitement que constitue l'extraction de lignes pour sélectionner des patchs contenant du texte, l'approche aléatoire semble être acceptable. Seulement, cette approche est très dépendante des caractéristiques des papyrus en question. En effet, s'ils ne contiennent que très peu de texte, la majorité des patchs extraits ne contiennent donc pas de texte et l'on peut s'attendre à des performances plus faibles, comme représenté sur le tableau. 3.2.

Ces premiers résultats font office de "preuve de concept". Avec 82% de vrais positifs et 6% de faux positifs dans le meilleur des cas et 72% de vrais positifs et 14% de faux positifs dans le pire des cas, nous sommes confiants dans la capacité d'une approche utilisant des réseaux de neurones siamois profonds à déterminer si deux patchs proviennent du même fragment. Maintenant, nous souhaitons déterminer si deux fragments proviennent du même papyrus. Pour cela, nous construisons des jeux de données d'entraînement dans lesquels les paires sont toutes constituées de patchs provenant de fragments différents. Si deux patchs proviennent de fragments qui appartiennent au même papyrus, alors ils sont considérés comme *similaires*, sinon, ils sont considérés comme *dissimilaires*.

## 3.5 Prédiction d'appairages de fragments

Nous avons montré dans la section 3.4.2 que notre approche permettait d'apprendre à déterminer si deux patchs proviennent du même fragment. Nous souhai-

tons maintenant déterminer si deux fragments proviennent du même document. Contrairement aux expériences précédentes, nous construisons ici des paires de patchs avec la connaissance de l'appartenance des fragments aux documents. C'est-à-dire que l'on ignore les paires de patchs provenant du même fragment, et on annote les paires de patchs extraits de deux fragments provenant du même document comme étant similaires. De cette manière, le modèle apprend effectivement à déterminer si deux patchs proviennent du même document. Nous mettons par la suite en place un protocole d'évaluation, et nous utilisons désormais toutes les données qui sont à notre disposition pour l'entraînement et l'évaluation de nos modèles.

### 3.5.1 Protocole d'évaluation

Parmi ces papyrus extraits de la collection de l'Université du Michigan, dont les détails sont donnés dans la Section. 3.3.1, beaucoup ne sont composés que d'un seul fragment (cf. Fig. 3.13) et ne peuvent donc pas être utilisés pour l'entraînement ni la validation. En effet, nous avons maintenant besoin de créer des paires de patchs dont chaque élément provient de fragments différents. Ces fragments seuls ne sont donc pas inclus dans le jeu de données. Au final, ce jeu de données est constitué de 4579 fragments pouvant être reconstruits en 1118 papyrus.

Nous utilisons aussi la base de données de la compétition *Hisfrag*, décrite dans la section 3.3.2. Cette base est constituée de documents anciens qui ne sont pas des papyrus, et qui ont été déchirés en fragments de manière synthétique. Sa taille est beaucoup plus importante (environ 100000 fragments qui peuvent être reconstruits en environ 17000 documents pour le jeu d'entraînement, et environ 20000 fragments pouvant être reconstruits en environ 2700 documents pour le jeu de test (cf. Fig 3.3)).

| | Hisfrag train | Hisfrag test | Michigan |
|---|---|---|---|
| Nb papyri | 17222 | 2732 | 1118 |
| Nb fragments | 101706 | 20019 | 4579 |
| mean frags/papy | 5.9 | 7.3 | 4.4 |
| std frags/papy | 9.7 | 2.9 | 6.4 |

TABLEAU 3.3 – Résumé des statistiques sur les bases *Michigan* et *Hisfrag*

Ces deux jeux de données ont des propriétés différentes. Ils ne sont pas constitués du même type de documents (ils ont des aspects très différents, cf. Figure. 3.21), le nombre de fragments par document est différent (cf. Tableau. 3.3 et Figure. 3.13) et les fragments du jeu *Hisfrag* sont créés artificiellement, alors que ceux du jeu *Michigan* sont "naturels". Par ailleurs, le fait que les fragments du jeu *Hisfrag* aient été créés artificiellement garantit la véracité de la vérité terrain, alors que les associations de fragments que l'on retrouve dans la collection de l'Université du Michigan

sont susceptibles de contenir des erreurs, car elles ont été faites par des humains. Il nous parait intéressant de comparer ces deux situations, d'un côté un jeu de données dont les annotations sont garanties, mais qui est synthétique, et de l'autre un jeu de données "naturel", dont les fragments sont authentiques, mais qui peut contenir des erreurs.

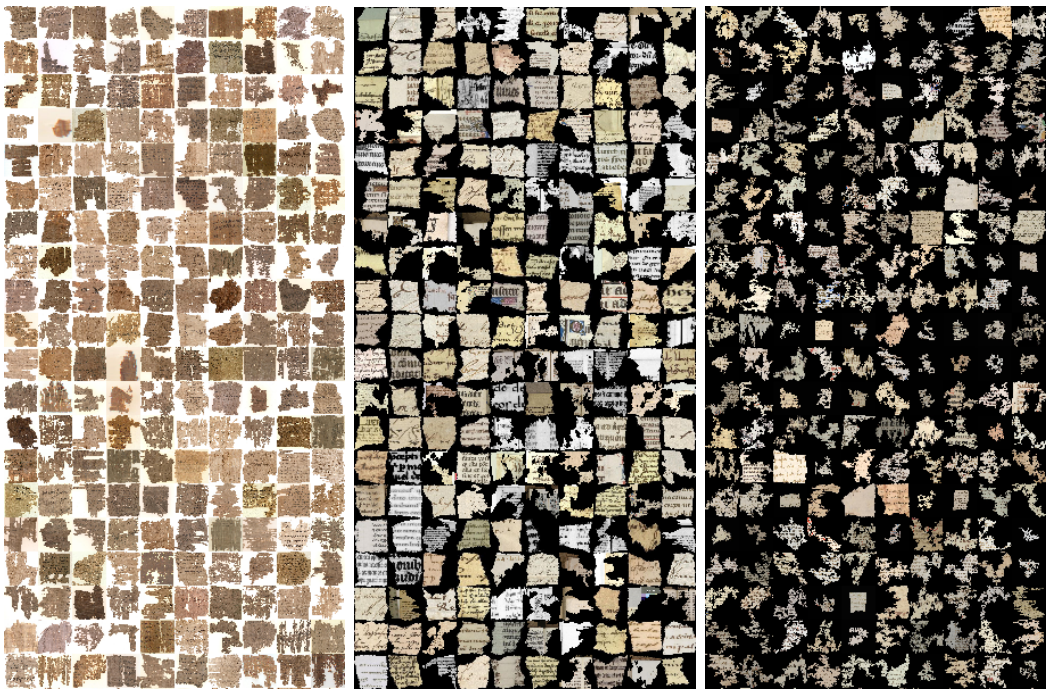


FIGURE 3.21 – Exemples de fragments provenant de la base Michigan (à gauche), de la base Hisfrag entraînement (au milieu) et de la base Hisfrag test (à droite).

En plus d'utiliser ces deux jeux de données de taille plus importante, nous choisissons de passer à des architectures plus communes pour les parties convolutives de nos réseaux siamois. Ainsi, nous construisons une architecture générique de réseau siamois dans laquelle on peut insérer directement des architectures tirées de réseaux état de l'art dans les branches. On utilise spécifiquement les parties convolutives des réseaux *VGG16* et *ResNet50* (cf. Section. 2.2.2), on a donc deux versions de notre réseau siamois que l'on peut comparer. Pour chacun des deux jeux de données, on entraîne les deux versions du réseau pendant 100 epochs, avec des batchs de taille 128, toujours avec l'optimiseur *AdamOptimizer*, avec un *learning-rate* dégressif de 0.001 à 0.00005.

Pour des raisons de temps de calcul, le calcul des métriques d'évaluation sur les jeux de tests est fait sur des sous-ensembles de 100 documents. Puisque toutes les combinaisons de paires de fragments sont évaluées lors de cette phase, le temps de calcul augmente de manière quadratique avec le nombre de documents. Par exemple, avec 100 documents sur la base *Hisfrag*, on a environ 800 fragments. On a extrait

5 patchs par fragments, on a donc 4000 patchs. On a donc un ordre de grandeur de $4000 \times 4000 = 16000000$ comparaisons. Calculer la similarité d'une paire prend environ 15 millisecondes, ce qui donne déjà plus de 6 heures de calculs.

### 3.5.2 Résultats

Les résultats sont recensés dans le Tableau. 3.4. On remarque des différences de performance relativement importantes entre les deux jeux de données, la version *VGG16* donne de meilleurs résultats sur *Hisfrag*, alors que la version *Resnet50* fonctionne mieux sur *Michigan*. De manière générale, les résultats sont moins bons sur la base *Michigan*, alors que les données peuvent sembler plus difficiles sur *Hisfrag* car les fragments du jeu de test sont créés de manière différente des fragments du jeu d'entraînement et ont des aspects très différents. Cet effet est certainement dû à la grande différence de quantité de données d'entraînement entre les deux bases. Comme on peut le voir sur le Tableau. 3.3, le jeu d'entraînement *Hisfrag* contient plus de dix fois plus de données que la base *Michigan* tout entière.

| | | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|
| Hisfrag | VGG16 | **0.67** | **0.87** | **0.75** | **0.92** |
| | Resnet50 | 0.61 | 0.83 | 0.71 | 0.82 |
| | Papy-S-Net | 0.52 | 0.75 | 0.6 | 0.86 |
| Michigan | VGG16 | 0.41 | 0.68 | 0.47 | 0.85 |
| | Resnet50 | **0.54** | 0.67 | **0.59** | **0.92** |
| | Papy-S-Net | 0.44 | **0.73** | 0.48 | 0.87 |

TABLEAU 3.4 – Comparaison des performances des trois architectures sur les deux bases de données. Ici, nous entraînons les modèles directement avec de l'information *niveau-document*. Les résultats sont calculés sur environ 800 fragments provenant du jeu de données Hisfrag, et environ 450 fragments provenant de la base de données Michigan

Le modèle basé sur *VGG16* et évalué sur le jeu *Hisfrag* atteint un score de *top-1 accuracy* moyen de 0.87. Cela veut dire qu'en moyenne, pour une requête donnée, la probabilité que le fragment le mieux classé soit valide (c'est-à-dire qu'il appartient au même document que le fragment requête) est de 87%. Pour mettre cela en perspective, la probabilité attendue que le fragment le mieux classé soit valide en choisissant aléatoirement est $\frac{7.3}{800} = 0.009125 \approx 0.9\%$. Avec 7.3 fragments par papyrus en moyenne dans le jeu de test de *Hisfrag* et les 800 fragments considérés dans ce test.

Les scores de *mean average precision*($mAP$) relativement faibles indiquent qu'il y a une quantité non négligeable de fragments non valides qui sont mieux classés que des fragments valides. Un taux de faux positifs élevé est attendu de par le nombre de comparaisons qui sont calculées. En effet, avec 800 fragments, on calcule

$800 \times 799 = 639200$ prédictions. Même en imaginant un modèle quasi parfait qui se tromperait en donnant un score élevé à des paires de fragments non valides dans 0.1% des cas, on s'attendrait toujours à $639200 \times 0.001 = 639.2$ faux positifs. C'est la raison pour laquelle on s'intéresse à la *top-1 accuracy* et à la *pr@k*, ces métriques donnent une meilleure idée de l'utilité des modèles dans le cas pratique des papyrologues cherchant à reconstruire les papyrus.

Afin d'avoir une interprétation correcte des métriques *pr@10* et *pr@100*, il faut se rappeler que le sous-ensemble du jeu *Hisfrag* (par exemple) que nous utilisons dans nos tests est constitué de 800 fragments. Ainsi, une requête comporte 799 comparaisons (on ne compare pas le fragment à lui-même). En moyenne, on s'attend dans ce cas à 7.3 comparaisons valides (cf. Tableau. 3.3). Étant donnée la définition de la métrique *pr@k* donnée dans la Section. 3.2.5, le score de *pr@k* peut avoir deux sens. Si $k$ est inférieur au nombre de fragments valides qu'il est possible de retrouver, on calcule la proportion de fragments valides retrouvés par rapport à $k$. Dans ce cas, un score de 1 signifie que les $k$ premiers fragments retrouvés sont tous valides, mais qu'il pourrait exister d'autres fragments valides classés plus loin. Si d'un autre côté $k$ est supérieur au nombre de fragments valides qu'il est possible de retrouver, on calcule la proportion des fragments valides retrouvés par rapport au nombre total de fragments pouvant être retrouvés. Dans ce cas, un score de 1 signifie que tous les fragments valides possibles ont été retrouvés parmi les $k$ premiers fragments, mais que des fragments non valides pourraient exister au sein de ces $k$ fragments.

En regardant les résultats présentés sur le Tableau. 3.4, le score de *pr@10* de 0.75 sur la base *Hisfrag* avec le modèle basé sur *VGG16* signifie que parmi les 10 fragments les mieux classés, on retrouve en moyenne 75% de tous les fragments valides. On peut avoir la même interprétation pour le score *pr@100* de 0.92, parmi les 100 fragments les mieux classés, on retrouve en moyenne 92% de tous les fragments valides. Cela veut aussi dire que parfois, des fragments valides obtiennent un score très faible.

La Figure. 3.22 illustre quatre exemples de requêtes sur le jeu de test de la base *Hisfrag*. Pour chaque colonne, le fragment requête est placé en haut de la colonne, et les fragments en dessous correspondent aux 10 fragments les mieux classés par rapport à ce fragment requête. D'après ces exemples, on voit que le modèle semble mieux fonctionner lorsque les fragments requêtes sont moins dégradés (colonnes 1 et 3). Cela n'est pas surprenant, car, comme on peut le voir sur la Figure. 3.21, le jeu d'entraînement de la base *Hisfrag* contient beaucoup moins de fragments très dégradés que le jeu de test. Le modèle a donc été entraîné sur des fragments peu dégradés alors que les évaluations que nous fournissons ici ont été faites sur un jeu de données très dégradé.

FIGURE 3.22 – Exemple de quatre requêtes. Chaque colonne correspond à une requête. Le fragment du haut est le fragment requête et les fragments du dessous sont les dix fragments ayant été classés avec les meilleurs scores, dans l'ordre décroissant. Les images des fragments ont été redimensionnées sans conserver leurs proportions dans cette visualisation.

# Conclusion

Après avoir dressé un état de l'art relatif à la reconstruction automatique de documents, nous avons vu dans ce chapitre les méthodes que nous avons développées pour calculer des distances entre des images de fragments de documents. Nos premières expériences sur l'architecture *Papy-S-Net* ont mis en évidence l'importance de sélectionner des patchs contenant du texte et ont confirmé la capacité des architectures de types siamoises à déterminer si deux patchs proviennent du même

fragment avec une précision dépassant 80%.

Après avoir défini comment calculer une distance entre fragments à partir d'une collection de distances entre patchs, nous avons mis en place notre protocole d'évaluation. Celui-ci consiste en la construction de requêtes dans lesquelles on compare un fragment à tous les autres fragments du jeu de données. Pour chaque fragment, on obtient donc une liste classée des fragments du plus proche au plus éloigné. Nous évaluons la pertinence des résultats de ces requêtes grâce à des métriques standards d'Information Retrieval (*mAP, top-1 accuracy, pr@k*). Ces nouvelles expériences ont été menées sur deux jeux de données de tailles plus significatives et en comparant les parties convolutives de modèles standards (*VGG16* et *ResNet50*).

Nos expériences montrent des performances qui diffèrent entre les deux jeux de données, de part leurs caractéristiques leurs tailles, mais aussi des différences de performances des modèles sur chaque base. Le modèle basé sur *VGG16* fonctionne mieux sur le jeu *Hisfrag* et le modèle basé sur *Resnet50* fonctionne mieux sur le jeu *Michigan*. Au-delà de cela, on obtient de manière générale des résultats plutôt satisfaisants dans toutes les configurations avec par exemple 87% de *top-1 accuracy* dans le meilleur des cas et 75% dans le pire des cas sur le jeu *Hisfrag*. Ces chiffres sont de 73% dans le meilleur des cas et 68% dans le pire des cas sur le jeu *Michigan*.

Les performances obtenues permettraient déjà aux papyrologues de gagner un temps précieux en limitant leurs recherches de fragments correspondant aux 10 ou 100 premiers plutôt qu'à l'entièreté de la base. Ces prédictions peuvent bien entendu être combinées avec des métadonnées pour filtrer plus encore les résultats.

Cependant, l'approche que nous avons décrite dans ce chapitre nécessite une relativement grande quantité de données annotées pour fonctionner. Il n'est pas réaliste de demander à des papyrologues d'annoter une grande partie de leurs bases afin d'obtenir des suggestions d'appairages sur cette même base. Dans le chapitre suivant, nous présentons notre deuxième contribution en explorant comment produire de telles suggestions d'appairages dans des contextes ou peu (voire pas du tout) de données annotées sont disponibles. Nous explorons l'apprentissage par transfert et la proposition au cœur de cette thèse, **l'apprentissage auto-supervisé.**

# Chapitre 4

# Stratégies d'entraînement en présence de peu de données

**Sommaire**

# Introduction

Utiliser une approche utilisant du *Deep Learning* nécessite souvent d'obtenir une masse suffisante de données annotées pour entraîner et évaluer les modèles développés. Le domaine de la reconstruction de fragments de documents ne déroge pas à la règle. Le jeu de données basé sur la collection *Michigan* que nous proposons contient une quantité raisonnable de fragments, mais elle est relativement loin des ordres de grandeur permettant de se rapprocher de modèles génériques ayant des performances acceptables sur différentes bases de données du même domaine que l'on peut retrouver avec la base *ImageNet* par exemple. La base de la compétition *Hisfrag* s'en rapproche, mais les fragments sont construits artificiellement via un algorithme de déchirures, ce qui peut introduire des biais liés à cet algorithme.

Jusqu'à présent, nous avons montré que notre approche permet de faire des prédictions sur une base de données si on a accès à une masse relativement grande de données d'entraînement de cette même base. Dans la pratique, les chercheurs en humanités numériques travaillent souvent sur des bases de relativement petite taille (cf. A). Par exemple, à l'issue de fouilles, des papyrologues peuvent avoir retrouvé quelques centaines de fragments de papyrus. Ils souhaitent ensuite étudier ce corpus, mais doivent dans un premier temps le reconstruire. C'est notamment le cas de la collection *Jouguet* du projet *GESHAEM*, qui est le sujet du Chapitre 5 de ce manuscrit.

Nous voulons pouvoir proposer des suggestions d'assemblages sur ces nouvelles données, cela en minimisant le travail d'annotation manuel. Nous explorons trois approches à cette fin :

— **Apprentissage par transfert** : On utilise des modèles entraînés sur d'autres bases de données de tailles conséquentes pour faire des prédictions sur la nouvelle base. Nous verrons qu'utiliser directement un modèle entraîné sur une autre base sur la nouvelle fournit des résultats peu performants. On peut alors demander aux experts d'annoter une petite partie de la nouvelle base afin de pouvoir faire du ré-entraînement. Nous verrons que cette approche améliore significativement les performances par rapport à l'approche *from scratch*. Naturellement, plus on utilise de données annotées pour le ré-entraînement, plus cette approche est performante. Mais plus on demande de travail manuel aux experts, moins l'automatisation est utile.
— **Apprentissage auto-supervisé** : On exploite l'information contenue dans les données non annotées pour entraîner des modèles. Cela se fait au travers de l'apprentissage d'une *tâche-prétexte* pendant l'entraînement, qui permettra au modèle de résoudre la *tâche-objectif*. On montrera qu'il est possible d'obtenir des performances compétitives sans avoir accès à aucune forme d'annotation des données ni aucun pré-entraînement. On peut combiner cette

approche avec un pré-entraînement sur une autre base annotée et encore améliorer les performances.

— **Sélection dans les batchs et augmentation de données** : Nous explorons la possibilité de combiner augmentation de données avec une approche sélectionnant dynamiquement les exemples difficiles d'un batch pendant le processus d'entraînement, de manière similaire au "minage de triplets" décrit dans la Section. 2.4.4.

L'approche *auto-supervisée* que nous proposons a fait l'objet d'une publication dans le journal *IJDAR* en 2021:

▷ [Pirrone et al. (2021)] Antoine Pirrone, Marie Beurton-Aimar, Nicholas Journet **Self-supervised deep metric learning for ancient papyrus fragments retrieval**, *International Journal on Document Analysis and Recognition (IJDAR 2021)*

## 4.1 Apprentissage par transfert

Une rapide introduction à l'apprentissage par transfert a été donnée dans la section 2.2.3. Pour rappel, l'idée de cette méthode est d'exploiter les caractéristiques qu'un modèle a pu apprendre à reconnaître grâce à une base d'entraînement pour entraîner le modèle sur de nouvelles données, ou pour une nouvelle tâche, de manière plus efficace. En effet, il parait pertinent de réutiliser, par exemple, les extracteurs de caractéristiques de bas niveau (détection de contours, de formes, etc.) entre plusieurs tâches, même si le domaine applicatif est très différent. Ainsi, il est aujourd'hui courant de pré-entraîner les modèles avec une grande base de données de type *ImageNet* car la quantité de données disponibles facilite l'apprentissage de ces extracteurs de caractéristiques. On peut ensuite choisir de *geler* certaines couches, le plus souvent les premières couches, et de ré-entraîner le modèle sur nos données. Le processus d'apprentissage peut ainsi se concentrer sur les caractéristiques de plus haut niveau spécifiques à nos données en exploitant des extracteurs de bas niveau robustes. On gagne au passage du temps de calcul.

Malheureusement, les expériences que nous avons conduites n'ont pas permis de faire converger nos modèles en utilisant des *VGG16* et *ResNet50* qui avaient été pré-entraînés sur *ImageNet*. La raison de cela est probablement que *ImageNet* est constituée d'un grand nombre de classes très différentes d'aspect (animaux, objets, véhicules ...) alors que nos jeux de données sont entièrement composés de documents. Ainsi, un modèle entraîné à reconnaître les classes d'*ImageNet* doit produire des extracteurs de caractéristiques très variés afin de capter la diversité des données. À l'inverse, les extracteurs de caractéristiques d'un modèle entraîné pour travailler sur un sous-domaine précis doivent être plus fins pour capter les détails qui permettent la discrimination entre les éléments du sous-domaine, qui se ressemblent globalement. En somme, dans notre cas les modèles semblent converger beaucoup mieux s'ils sont entraînés en partant de poids aléatoires plutôt que de poids issus d'un pré-entraînement avec *ImageNet*.

On se place donc dans une configuration plus proche du sous-domaine de l'apprentissage par transfert qu'est l'*adaptation de domaine*. Le concept est le même, mais on se restreint à utiliser des bases de données d'un domaine proche du nôtre pour le pré-entraînement. Pour nos expériences, nous avons utilisé le jeu de données *Hisfrag* comme source de pré-entraînement, car elle est de bien plus grande taille que la base *Michigan*.

### 4.1.1 Transfert direct

L'objectif de ce chapitre est d'évaluer ce qu'il est possible de faire dans le contexte où l'on ne dispose que de peu de données d'entraînement pour la tâche que l'on souhaite résoudre, comme dans le cas de la base *GESHAEM* que nous étudions dans le

Chapitre. 5. Nous regardons dans un premier temps s'il est possible d'utiliser directement un modèle entraîné sur une autre base relativement similaire pour obtenir des prédictions sur notre tâche. Pour cela, nous réutilisons les modèles entraînés sur la base *Hisfrag*, dont les résultats sont donnés dans le Tableau. 3.4. On effectue des prédictions sur le jeu de test de la base *Michigan* directement à partir des modèles entraînés sur *Hisfrag*.

Les résultats de cette expérience sont donnés dans le Tableau. 4.1. On peut voir que les performances sont plutôt mauvaises, avec des scores environ deux fois plus faibles que lorsque les modèles étaient entraînés directement sur la base *Michigan* (cf. 3.4). Par exemple, la meilleure *mAP* que nous obtenons ici est de 0.26, alors que la meilleure *mAP* obtenue sur *Michigan* dans les expériences du Tableau. 3.4 est de 0.54.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | 0.21 | 0.37 | 0.29 | 0.59 |
| Resnet50 | **0.26** | **0.47** | **0.31** | **0.63** |

TABLEAU 4.1 – Évaluation des modèles pré-entraînés sur la base *Hisfrag* sans ré entraînement. Évaluation sur le jeu de test de la base *Michigan*

Cette première approche ne semble donc pas satisfaisante. Nous explorons ensuite les possibilités offertes par le ré-entraînement de modèles avec différentes quantités de données annotées.

### 4.1.2 Ré-entraînement

Comme précédemment, nous partons des modèles pré-entraînés sur la base *Hisfrag*, mais cette fois, nous rajoutons une étape de ré-entraînement (*fine-tuning*) en utilisant des données annotées de la base objectif, la base *Michigan*. Aucune couche n'est gelée dans les expériences suivantes.

On souhaite bien sûr n'avoir à annoter qu'une petite quantité de données pour limiter le travail humain. Voyons ce que 50 documents annotés peuvent apporter lors d'un ré-entraînement. Pour référence, si l'on entraîne à partir de zéro nos modèles avec seulement 50 documents de la base *Michigan*, soit environ 220 fragments [1], les performances sont extrêmement mauvaises comme on peut le voir sur le Tableau. 4.2.

En partant d'un modèle pré-entraîné sur *Hisfrag* par contre, l'utilisation des 50 documents annotés améliorent sensiblement les performances sur le jeu de test de la base *Michigan* par rapport à l'expérience précédente (cf. Tableau. 4.1). Comme on le voit sur le Tableau. 4.3, on obtient dans le meilleur des cas une *mAP* supérieure

1. Il est à noter que 200 fragments représentent déjà un travail d'annotation manuel conséquent.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | 0.07 | 0.07 | 0.06 | 0.29 |

TABLEAU 4.2 – Le modèle est entraîné avec seulement 50 papyrus de la base Michigan et évalué sur l'ensemble du jeu de test de la base Michigan

de 8 points. Cela signifie, et c'était attendu, que si on a accès à un modèle déjà entraîné sur une autre base, annoter une petite quantité de données et faire un ré-entraînement peut déjà permettre d'obtenir de bien meilleurs résultats que sans pré-entraînement.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | **0.34** | **0.60** | **0.41** | **0.79** |
| Resnet50 | 0.31 | 0.56 | 0.37 | 0.72 |

TABLEAU 4.3 – Évaluation des modèles pré-entraînés sur la base *Hisfrag* et ré-entraînés avec 50 papyrus de la base *Michigan*. Évaluation sur le jeu de test de la base *Michigan*.

On s'intéresse aussi aux performances que l'on peut atteindre en ré-entraînant avec une grande quantité de données annotées. A quantité de données d'entraînement / ré-entraînement égales, on pourrait s'attendre à de meilleures performances avec ré-entraînement que sans, mais il se trouve que l'on observe le contraire dans notre cas. Sur le Tableau. 4.4, les modèles sont ré-entraînés à partir des modèles entraînés sur la base *Hisfrag* avec 1000 papyrus de la base *Michigan*, soit autant que dans les expériences du chapitre précédent dont on peut retrouver les résultats dans le Tableau. 3.4. Les résultats sont significativement améliorés par rapport à un pré-entraînement avec seulement 50 papyrus, mais de manière contre intuitive, ils sont moins bons que sans aucun pré-entraînement. En effet, dans le premier cas, on obtient une $mAP$ de $0,45$, et dans le second une $mAP$ de 0.54 soit presque 10 points de différence. On peut aussi noter que passer de 50 à 1000 papyrus de ré-entraînement a eu un effet marginal sur le modèle basé sur *VGG16*, mais un effet significatif sur le modèle basé sur *Resnet50*. Une étude plus poussée serait nécessaire pour comprendre l'origine de ces différences. Cette approche nécessitant cependant toujours un travail d'annotation manuel, nous avons décidé de nous concentrer sur une autre méthode, ne nécessitant aucune donnée annotée. Comme nous allons le voir dans la section suivante, cette méthode permet d'obtenir des résultats plus exploitables de par la quantité réduite de travail humain nécessaire à sa mise en place.

## 4.2 Apprentissage *auto-supervisé*

Les résultats présentés jusqu'à présent présupposent l'accès à une certaine quantité de données annotées, c'est-à-dire une vérité terrain *niveau-document* (on sait

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | 0.35 | 0.62 | 0.42 | 0.81 |
| Resnet50 | **0.45** | **0.74** | **0.49** | **0.86** |

TABLEAU 4.4 – Évaluation des modèles pré-entraînés sur la base *Hisfrag* et ré-entraînés avec 1000 papyrus de la base *Michigan*. Évaluation sur le jeu de test de la base *Michigan*.

quels fragments appartiennent à quels documents). C'est le cas pour le pré entraînement d'un modèle qui servira de base pour un ré-entraînement ou pour l'apprentissage d'un modèle lui-même. Cependant, dans une situation réelle, les papyrologues ou autres historiens des documents peuvent travailler sur des fragments de documents retrouvés récemment, ou qui n'ont pas encore été étudiés. Pour être utiles, les méthodes que nous proposons doivent nécessiter le moins possible de travail d'annotation manuelle, idéalement, aucun. Nous explorons ainsi la possibilité d'entraîner des modèles dans un contexte où *aucune* information *niveau-document* n'est disponible. Pour cela, nous proposons une approche *auto-supervisée*.

### 4.2.1 Proposition d'une stratégie d'apprentissage auto supervisée

Comme expliqué dans la section 2.2.4, on appelle *tâche-prétexte* la tâche sur laquelle le modèle s'entraîne, et on appelle *tâche-objectif* la tâche que nous souhaitons réellement résoudre. Toute la puissance de cette approche vient du fait que la *tâche-prétexte* est conçue de telle manière que pour apprendre à la résoudre, le modèle doit obtenir une compréhension des données qui sera utile pour résoudre la *tâche-objectif*. Et la *tâche-prétexte* peut être conçue à partir des données elles-mêmes, sans annotation.

Ici, notre *tâche-prétexte* est de déterminer si deux patchs proviennent du même fragment. Notre *tâche-objectif* est de déterminer si deux fragments proviennent du même document. Pour construire le jeu de données pour la *tâche-prétexte*, on annote automatiquement les paires de patchs provenant du même fragment comme *similaires* et les paires de patchs provenant de fragments différents comme *dissimilaires*. Comme on peut le voir sur la Figure. 4.1, cela a pour effet de bord de mal annoter une partie des paires.

On voit sur la Figure. 4.1 que les patchs 1 et 2 proviennent du même fragment et sont donc annotés *similaires* (flèche verte), idem pour les patchs 3 et 4. Les patchs 1 et 3 ne proviennent pas du même fragment et sont donc annotés *dissimilaires* alors que les deux fragments desquels ils proviennent appartiennent au même document. C'est une mauvaise annotation. Les patchs 4 et 6 sont quant à eux correctement annotés comme *dissimilaires*, car ils proviennent de deux documents différents.

Comme expliqué dans la section 3.2.3, nous construisons des batchs de paires de

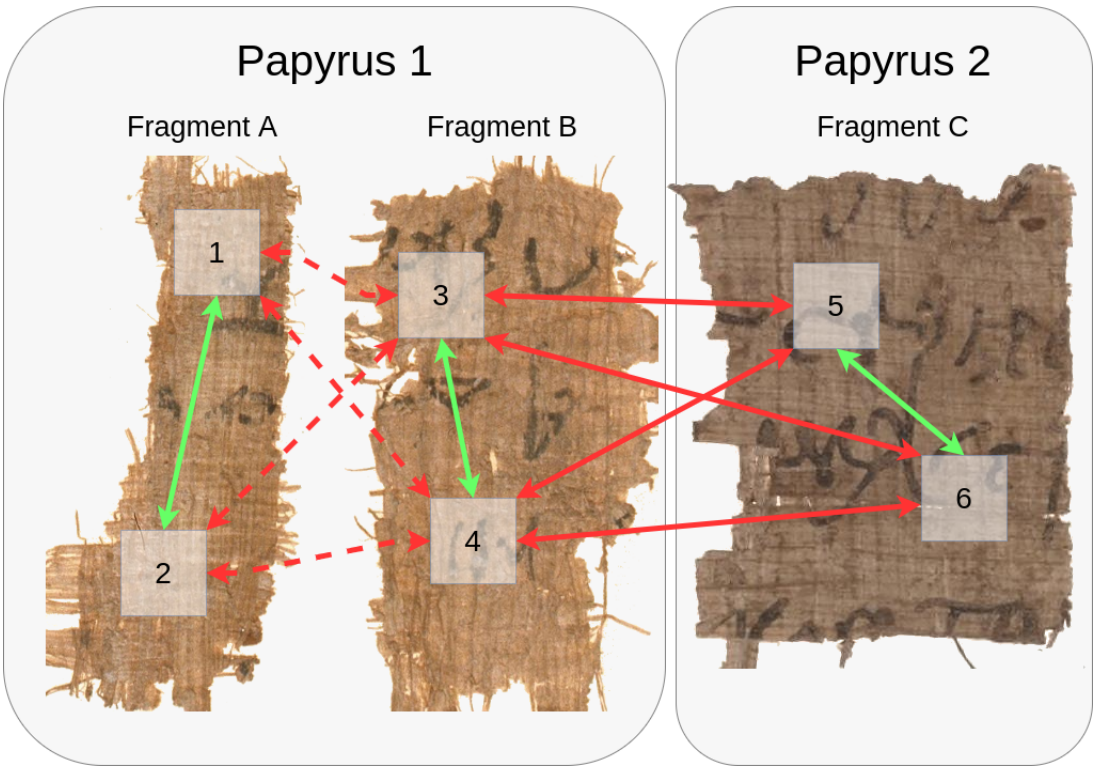



FIGURE 4.1 – Construction de paires de patchs pour l'entraînement de la *tâche-prétexte*. Les flèches vertes représentent des paires *similaires*, les flèches rouges des paires *dissimilaires* et les flèches rouges pointillées des paires *dissimilaires* mal annotées.

patchs équilibrés pour l'entraînement. Pour ce faire, on sélectionne un patch aléatoire dans la collection de tous les patchs extraits, puis un patch *similaire* aléatoire une fois sur deux et un patch *dissimilaire* aléatoire l'autre fois. Lorsque l'on sélectionne le patch *dissimilaire*, il est possible de sélectionner un patch provenant du même document, mais pas du même fragment. C'est une mauvaise annotation. Calculons une estimation statistique du nombre de paires mal annotées qui seront ainsi sélectionnées.

Prenons l'exemple de 10000 documents provenant du jeu de données d'entraînement de *Hisfrag*. D'après le Tableau. 3.3, on aura en moyenne $10000 \times 5.9 = 59000$ fragments, car en moyenne, chaque document est composé de 5.9 fragments. Si on décide d'extraire 5 patchs par fragment, on obtient en au total en moyenne $5 \times 59000 = 295000$ patchs et $5 \times 5.9 = 29.5$ patchs par document. La probabilité de sélectionner un patch provenant du même document, mais ne provenant pas du même fragment est donc la suivante :

$$P_{mis} = \frac{29.5 - 5}{295000 - 5} = 8.3e - 05$$

Avec une taille de batch de 128, on sélectionne 64 paires *dissimilaires*, donc

lorsque l'on crée un batch, la probabilité qu'il contienne une paire mal annotée est :

$$P_{mis-batch} = P_{mis} \times 64 \approx 0.005$$

Puisque cette probabilité est très faible, nous faisons l'hypothèse que cela devrait peu impacter l'entraînement dans ces conditions. Cependant, cette probabilité grandit si le nombre de documents diminue. Par exemple, avec une base de seulement 500 documents et 3 fragments par documents en moyenne, on obtient par les mêmes calculs une probabilité d'environ 0.09 qu'un batch de taille 128 contienne une paire mal annotée. C'est un effet qu'il faut garder à l'esprit lorsque l'on travaille en *auto-supervisé* avec des petits jeux de données.

### 4.2.2 Apprentissage *auto-supervisé* avec ou sans ré-entraînement

Pour rappel, nous appelons *niveau-fragment* l'information de l'appartenance des **patchs** aux **fragments** et *niveau-document* l'information de l'appartenance des **fragments** aux **documents**.

Nous entraînons chaque modèle *from scratch* avec seulement de l'information *niveau-fragment*, en évaluant avec de l'information *niveau-document* comme dans les expériences du Chapitre. 3. L'évaluation est faite de la même façon que dans le chapitre précédent, seulement, les modèles n'ont jamais eu accès à de l'information *niveau-document* pendant l'entraînement. Avec ces expériences, on évalue la capacité des modèles à résoudre la *tâche-objectif* à partir de la *tâche-prétexte*. Le Tableau. 4.5 contient les résultats de ces expériences. Les modèles ont été entraînés avec la même quantité de données que pour le Tableau. 3.4, soit 80% de chaque base (soit environ 80000 fragments pour *Hisfrag* et 3600 fragments pour *Michigan*), les 20% restants sont réservés aux jeux de validation et de test (soit environ 20000 fragments pour *Hisfrag* et 900 fragments pour *Michigan*).

| | | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|
| Hisfrag | VGG16 | 0.38 | **0.73** | 0.47 | 0.72 |
| | Resnet50 | **0.41** | 0.51 | **0.48** | **0.78** |
| Michigan | VGG16 | **0.38** | **0.61** | **0.43** | **0.84** |
| | Resnet50 | 0.30 | 0.43 | 0.39 | 0.76 |

TABLEAU 4.5 – Évaluation des modèles entraînés en mode *auto-supervisé*. Les modèles ont été entraînés avec seulement de l'information niveau-fragment et évalués avec de l'information niveau-document sur 100 documents.

Il n'est pas surprenant d'obtenir de moins bonnes performances lorsque l'entraînement est fait en mode *auto-supervisé* par rapport aux entraînements du chapitre précédent dans lesquelles les modèles sont entraînés directement sur la *tâche-objectif*.

En *auto-supervisé*, la *mAP*, la *top-1 accuracy* et la *pr@10* sont plus basses de 0.19 en moyenne. La *pr@100* est quant-à-elle plus basse de 0.05 en moyenne.

Il est cependant intéressant de remarquer que l'approche *auto-supervisée* obtient de légèrement meilleures performances que l'approche *apprentissage par transfert* ré-entraînée avec 50 documents (cf. Figure. 4.2). En effet, cette dernière nécessite d'avoir à disposition un modèle pré-entraîné ainsi que 50 documents annotés, alors que l'approche *auto-supervisée* est moins contraignante, car elle ne nécessite ni pré-entraînement, ni aucune donnée annotée. On peut aussi remarquer encore une fois que les meilleurs résultats sont obtenus sur la base *Hisfrag* grâce à sa grande quantité de données, mais l'écart avec les résultats sur la base *Michigan* est ici plus petit.

Nous évaluons aussi s'il est intéressant de combiner pré-entraînement et apprentissage *auto-supervisé*. Comme précédemment, nous ré-entraînons un modèle pré-entraîné sur *Hisfrag*, mais cette fois-ci en utilisant seulement de l'information *niveau-fragment*. On peut voir sur le Tableau. 4.6 que les performances obtenues sont très proches de celles obtenues lors du ré entraînement utilisant 1000 documents avec de l'information *niveau-document* (cf. Tableau 4.4). Il est donc bien plus intéressant dans ce cas de ré-entraîner en mode *auto-supervisé*, car cela évite d'annoter un grand nombre de données.

| | | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|
| Michigan | VGG16 | 0.32 | 0.32 | 0.38 | 0.78 |
| | Resnet50 | **0.43** | **0.75** | **0.47** | **0.84** |

Tableau 4.6 – Évaluation de modèles ré-entraînés en mode *auto-supervisé* à partir de modèles pré-entraînés sur Hisfrag

La Figure. 4.2 synthétise les performances des différentes expériences présentées jusqu'à présent sur le jeu de test de la base *Michigan*. Sur cette figure, la "*Baseline*" correspond aux modèles entraînés à partir de zéro avec toute l'information *niveau-papyrus* disponible. Cela correspond en quelque sorte à un maximum théorique que peut atteindre notre méthode dans des conditions idéales. La catégorie "Michigan-normal-50" correspond à la même expérience, mais avec seulement 50 documents d'entraînement. La catégorie "Pas de ré entraînement" correspond à l'utilisation directe d'un modèle entraîné sur *Hisfrag* pour faire des prédictions sur le jeu de test de *Michigan*. Les catégories "Ré-entraînement-50" et "Ré-entraînement-1000" correspondent aux modèles qui ont été ré-entraînés à partir d'un modèle entraîné sur *Hisfrag*, en utilisant respectivement 50 et 1000 documents annotés de *Michigan* pour le ré entraînement. Enfin, les catégories "Auto-supervisé" et "Auto-supervisé ré entraînement" correspondent aux modèles entraînés en mode *Auto-supervisé*, premièrement partir d'un modèle pré-entraîné, puis en partant d'un modèle pré-entraîné sur *Hisfrag*, et ré-entrant en utilisant toutes les données non annotées disponibles de *Michigan*, en mode *Auto-supervisé*.

La conclusion la plus intéressante que l'on peut tirer des résultats présentés sur la Figure. 4.2 est que l'approche *auto-supervisée* présente des performances compétitives par rapport à l'approche ré-entraînement. Toujours en gardant à l'esprit que l'on souhaite minimiser le travail manuel d'annotation, l'approche la plus pertinente semble être celle qui combine ré-entraînement et apprentissage *auto-supervisé*. On obtient dans ce cas des performances très proches du cas dans lequel on ré-entraîne avec 1000 documents annotés, alors que ce premier cas ne nécessite aucune annotation sur le jeu de données sur lequel on souhaite produire des suggestions, contrairement au deuxième cas.

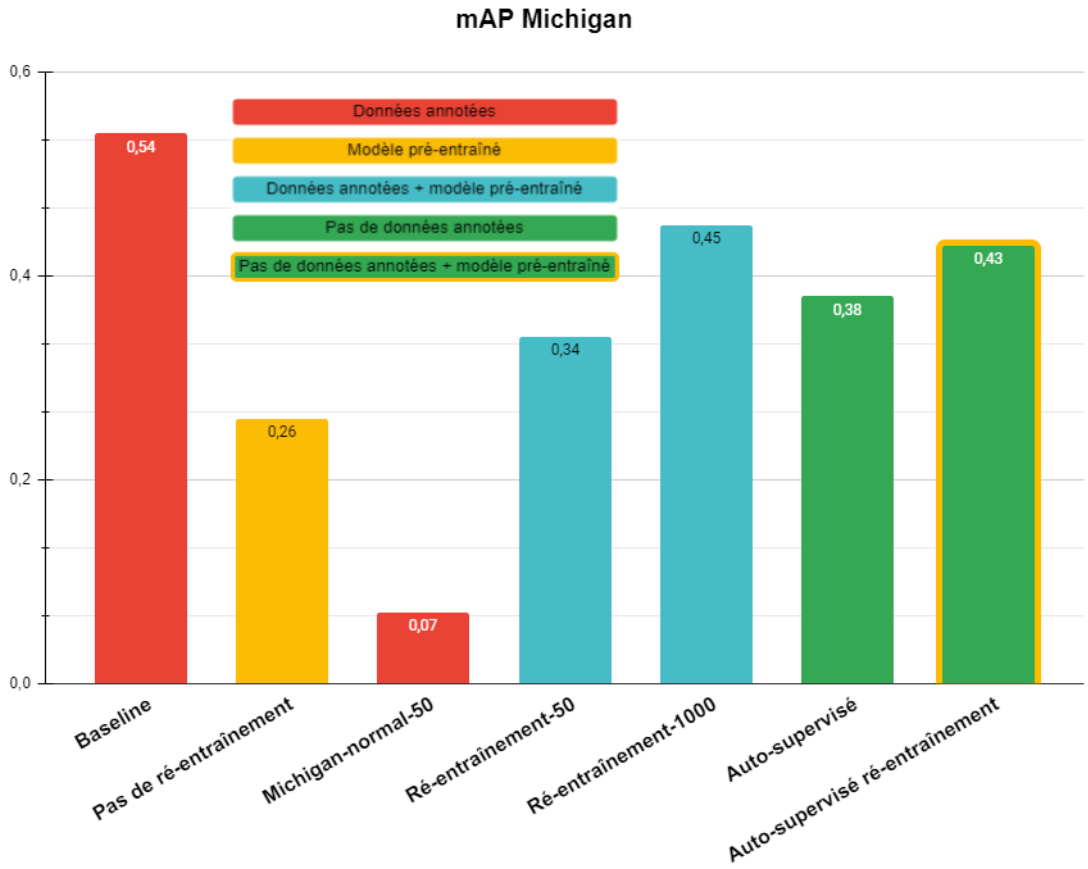



FIGURE 4.2 – Comparaison des performances des différentes approches testées jusqu'à présent. En rouge, les méthodes nécessitant des données annotées. En jaune, les méthodes nécessitant un modèle pré-entraîné. En bleu, les méthodes nécessitant un modèle pré-entraîné ainsi que des données annotées. En vert, les méthodes ne nécessitant pas de données annotées. En vert et jaune les méthodes ne nécessitant pas de données annotées et nécessitant un modèle pré-entraîné.

## 4.3 Approche *auto-supervisée* appliquée à de petites bases de documents

Plaçons-nous dans le contexte de chercheurs travaillant sur un corpus de fragments qui vient d'être découvert. Nous avons vu que réutiliser directement un modèle entraîné sur d'autres données donnait des résultats peu satisfaisants. Nous avons aussi vu que faire de l'apprentissage auto-supervisé avec de relativement grandes quantités de données (au moins 1000 documents, et donc plus de 4000 fragments au total) donnait des résultats intéressants. Dans le cadre des documents historiques, 1000 documents forment déjà un corpus plutôt grand, la plupart du temps, les tailles de corpus sur lesquels les chercheurs travaillent sont beaucoup plus petites. Cependant, même avec seulement 200 fragments, le processus de reconstruction peut être long et laborieux à faire manuellement. Nous souhaitons évaluer notre approche auto-supervisée sur un petit nombre de fragments afin de déterminer s'il est possible de construire des modèles ad-hoc pour chaque corpus donné. Autrement dit, est-il possible d'obtenir de bonnes prédictions d'appairages sur un corpus donné de petite taille (quelques centaines de fragments), en entraînant un modèle seulement sur ce corpus, en mode auto-supervisé.

Ici, nous allons donc entraîner nos modèles avec toutes les données disponibles d'un corpus réduit et faire des prédictions d'appairages sur ce même corpus (cf. Figure. 4.3). C'est un cas d'usage assez particulier, car on s'éloigne des objectifs classiques d'un modèle d'apprentissage automatique, qui est d'apprendre à généraliser pour prédire correctement sur de nouvelles données (cf. 2.3). Ici, on souhaite au contraire que le modèle "apprenne les données", ou plutôt qu'il apprenne à reconnaître des patchs provenant du même **document**, mais en ne lui montrant en tant que vérité terrain positive que des patchs provenant des mêmes **fragments**. Puisque nous voulons prédire sur les données avec lesquelles l'entraînement a été fait, nous n'avons pas besoin de nous préoccuper du problème du sur-apprentissage.

En résumé, on cherche à annoter automatiquement le corpus via un processus d'apprentissage *auto-supervisé*. Lors de l'entraînement, le modèle apprend à déterminer si deux patchs proviennent du même fragment et lors de l'inférence, on utilise ce modèle pour déterminer si deux patchs provenant de deux fragments différents proviennent en fait du même document.

Il faut prendre en compte deux "forces" qui travaillent l'une contre l'autre. D'un côté, plus on utilise de données pour l'entraînement, meilleures devraient être les prédictions. De l'autre, plus il y a de fragments à analyser, plus il est difficile de produire des appairages pertinents. Le nombre de paires de fragments augmentant de manière quadratique avec le nombre de fragments, les chances de tomber sur des faux positifs augmentent de la même manière.

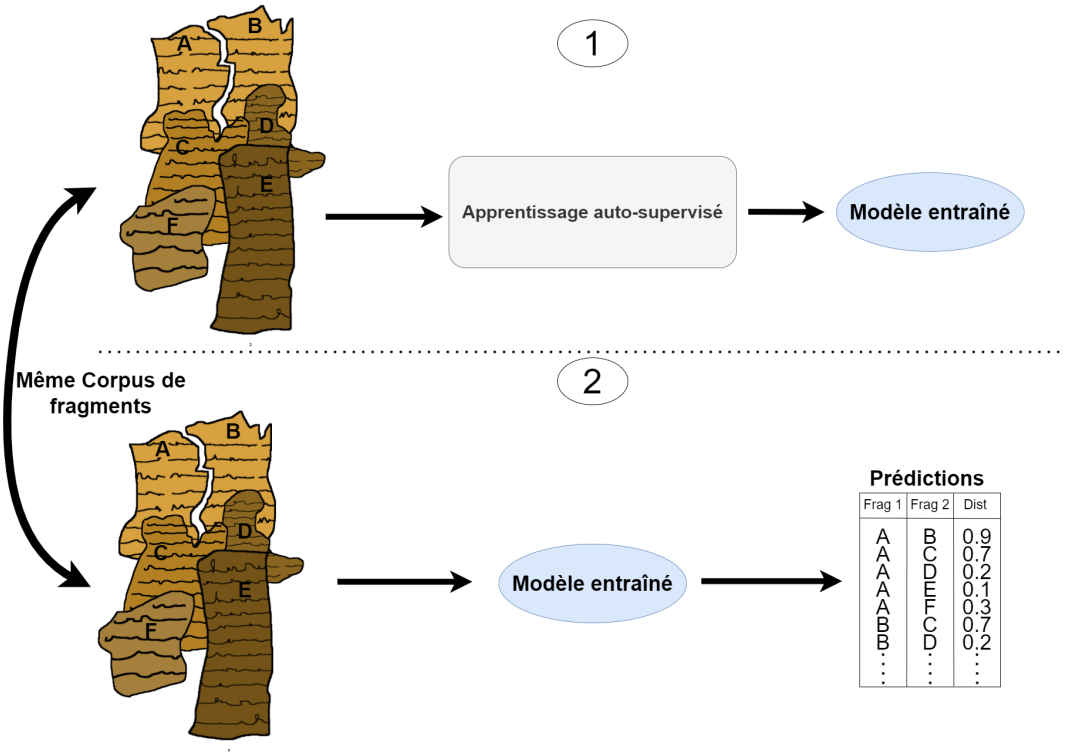
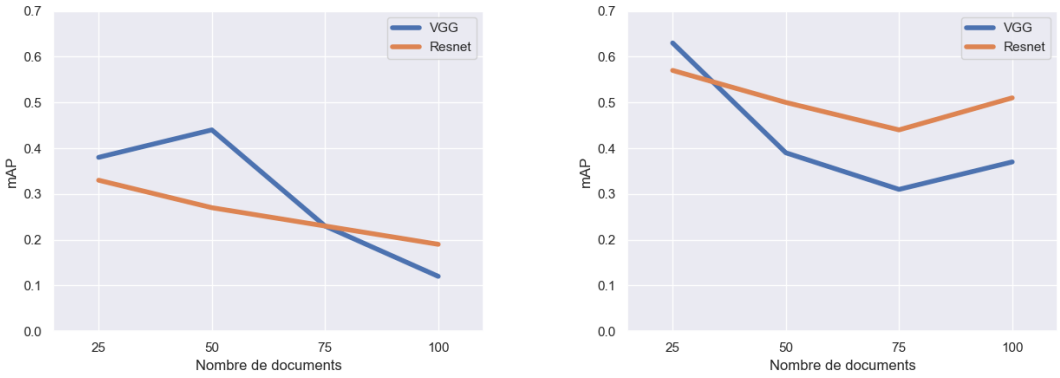



FIGURE 4.3 – Un modèle est entraîné par apprentissage *auto-supervisé* sur un corpus donné. On calcule ensuite des prédictions d'appairages des fragments de ce même corpus grâce au modèle entraîné précédemment.

Une partie des résultats est représentée sur la Figure. 4.4 et la totalité des résultats avec toutes les métriques d'évaluation utilisées peuvent être consultés dans les tableaux 4.4.a et 4.4.b de l'Annexe B.

On observe sur la Figure. 4.4 le phénomène décrit plus tôt. Avec peu de données, le modèle ne peut pas devenir très performant, mais la tâche d'association est relativement facile car la probabilité d'erreurs est faible. On remarque que l'augmentation de la difficulté de la tâche n'évolue pas à la même vitesse que l'augmentation des performances du modèle en fonction de la quantité de données. Par exemple dans le cas du jeu de données *Michigan*, on voit sur la Figure. 4.4a que le modèle est beaucoup moins performant en terme de mAP avec 100 documents qu'avec 25 alors qu'il a eu quatre fois plus de données d'entraînement. Pour le jeu de données *Hisfrag*, on voit sur la Figure. 4.4b que les performances baissent aussi avec l'augmentation de la quantité de données jusqu'à atteindre 75 documents, puis augmentent avec 100 documents.

On s'attend en effet à ce que la tendance s'inverse à partir d'un certain seuil de quantité de données aux vues des performances que l'on a pu obtenir dans les expériences précédentes. Ce seuil n'a par contre pas de raison d'être le même pour




(a) Michigan (b) Hisfrag

FIGURE 4.4 – Cas d'utilisation avec des données de la base Michigan (à gauche) et Hisfrag (à droite). Chaque modèle est entraîné en mode *auto-supervisé* et évalué sur les mêmes données qu'à l'entraînement.

les deux jeux de données. En effet, une différence majeure entre ces deux jeux de données, outre le fait qu'ils soient constitués de types de documents différents et que les fragments de *Hisfrag* sont synthétiques, est le nombre de fragments par document en moyenne. Pour rappel, les documents du jeu de données *Michigan* sont composés en moyenne de 4.4 fragments alors que ceux du jeu de données *Hisfrag* sont composés en moyenne de 5.9 fragments (cf. Tableau. 3.3). Cela signifie que pour le même nombre de documents pour ces tests, *Hisfrag* aura plus de données d'entraînement et de test que *Michigan*.

Il est intéressant de noter que dans la plupart des expériences présentées dans les sections précédentes, les modèles basés sur *Resnet50* semblent proposer de meilleures performances que ceux basés sur *VGG16* lorsque beaucoup de données d'entraînement sont disponibles, et inversement. On observe la même chose ici. Cela peut s'expliquer par le fait que les deux modèles ont des profondeurs très différentes, 16 couches pour *VGG16* et 50 couches pour *Resnet50*, en plus du mécanisme des connexions résiduelles décrit dans la Section. 2.2.2. Comme discuté dans la Section. 2.3, la *capacité* d'un réseau de neurones définit en partie ses aptitudes à généraliser la tâche en fonction de la quantité de données d'entraînement disponible. Un réseau ayant une capacité d'apprentissage (c'est-à-dire son nombre de paramètres) trop élevée par rapport à l'entropie du jeu d'entraînement (c'est-à-dire la quantité d'information qu'il contient) aura tendance à apprendre les données plutôt que de généraliser. Dans le cas inverse, soit une capacité d'apprentissage trop faible par rapport à l'entropie du jeu de données, le modèle est obligé de généraliser, mais il n'a peut-être pas assez de capacité pour capturer l'entièreté de la complexité de la tâche. Ici, on remarque qu'en fonction de la quantité de données que nous rendons disponible pour l'entraînement des modèles, l'un ou l'autre de ces effets se manifeste.

## 4.4 Sélection dans les batchs et augmentation de données

Pour essayer d'améliorer les résultats, nous proposons d'étudier et de combiner deux stratégies qui peuvent s'ajouter aux méthodes qu'on a présentées jusqu'à présent : l'augmentation de données et la sélection dans les batchs. L'augmentation de données est un processus très courant utilisé en apprentissage automatique que nous n'avons pas exploité dans nos expériences précédentes. Nous nous intéressions ici surtout à sa combinaison avec le processus de sélection dans les batchs.

### 4.4.1 Augmentation de données

Dans un contexte d'apprentissage automatique, le processus d'augmentation de données consiste en l'introduction de petites variations dans les données originales afin de rendre le modèle entraîné plus robuste ou invariant à ces transformations. Cela se fait en dupliquant certaines (ou toutes les) images de la base de données et en appliquant des transformations à ces images dupliquées. Traditionnellement, les images sont modifiées en utilisant des transformations comme des rotations, changements de luminosité/contraste, injection de bruit ou rognage (Goodfellow et al. (2016)).

L'idée est que certaines variations existent et sont susceptibles d'apparaître en production, mais n'ont pas nécessairement été capturées lors de l'acquisition des données d'entraînement, ou en trop faible quantité. Par exemple, si un jeu de données d'entraînement a été capturé dans des conditions de luminosité homogènes, il peut être utile de faire de l'augmentation de données en introduisant des variations de luminosité afin que le modèle entraîné soit robuste à ce type de variations lorsqu'il sera utilisé en production, où les conditions de luminosité seront peut-être moins homogènes.

Dans des cas où il est extrêmement difficile d'obtenir des données annotées, il est parfois envisageable de construire des jeux de données de manière entièrement synthétique. Cocosco et al. (1997) proposent à ce sujet une approche pour générer des images *IRM*[2] de cerveaux dont les caractéristiques peuvent varier en fonction de ce que l'on souhaite étudier. On peut aussi vouloir enrichir un jeu de données en générant des données synthétiques réalistes. Par exemple, Chen et al. (2020) proposent de simuler des corruptions de signaux réalistes sur des données médicales qui correspondent à des artéfacts qui sont susceptibles d'apparaître sur des images *IRM*.

Dans le contexte des documents, le logiciel *DocCreator*[3] (Journet et al. (2017))

2. Imagerie par Résonance Magnétique
3. https://doc-creator.labri.fr/

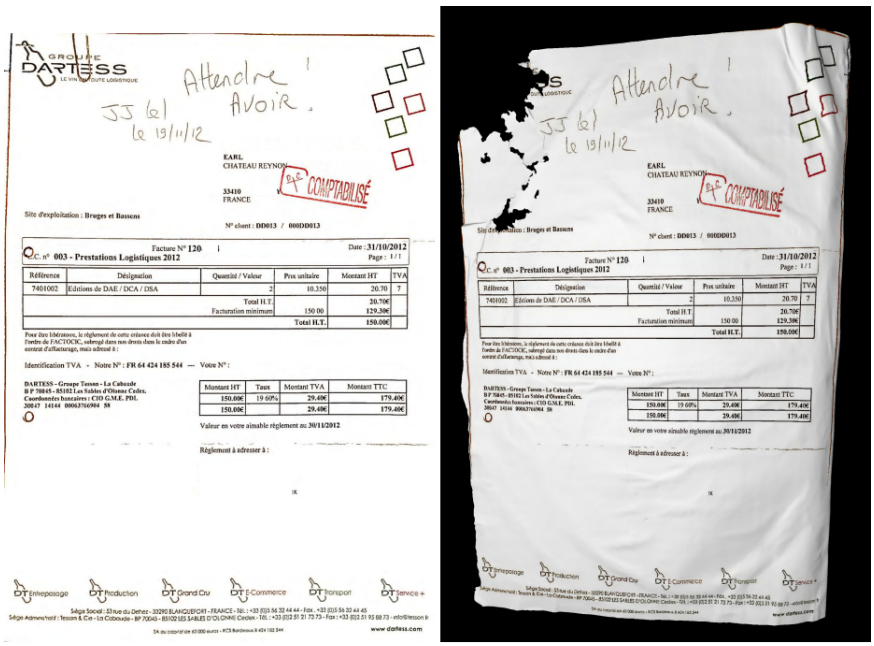


FIGURE 4.5 – Exemple d'une dégradation réaliste de document générée par *DocCreator*. Source : Journet et al. (2017).

permet la génération de dégradations réalistes sur des documents. Ces dégradations comprennent entre autres des déchirures, des plis, de l'effacement d'encre ou encore l'ajout de caractères fantômes (cf. Fig. 4.5). Le logiciel permet aussi la génération de documents synthétiques entiers basée sur des images réelles. On peut aussi citer le logiciel *DocEmul*, qui propose sensiblement le même service que *DocCreator* (Capobianco et Marinai (2017)). Il faut aussi bien sûr citer les approches exploitant les *Generative Adversarial Networks (GAN)*, qui sont depuis quelques années de plus en plus présentes. Par exemple, Pondenkandath et al. (2019) proposent d'utiliser des GAN pour générer de grandes quantités de documents historiques synthétiques par transfert de style à partir de documents anciens, avec le contenu de documents électroniques (type *pdf*).

En première intention, nous souhaitons isoler une transformation (ou une combinaison de transformations) qui améliore les performances de nos modèles. Pour cela, nous avons évalué l'impact de différentes combinaisons de transformations disponibles dans *DocCreator* (Journet et al. (2017)). La Figure. 4.7 illustre 5 dégradations parmi les 10 possibles que le logiciel DocCreator est en mesure de générer.

Pour chaque expérience, un modèle basé sur *VGG16* est entraîné à partir de zéro (c'est-à-dire sans faire de ré-entraînement) en mode *auto-supervisé* sur la base *Michigan* et les mêmes modalités de séparation en jeux d'entraînement/validation/test qu'auparavant ont été utilisées. Cependant, ici, la quantité de données est doublée, car chaque fragment est dupliqué et une ou plusieurs transformations sont appliquées

à ce fragment dupliqué. Une expérience consiste à choisir une stratégie d'augmentation de données (ou une combinaison), on apprend sur ce jeu de données, et on évalue sur des données réelles. Pour chaque expérience, on calcule la *mAP*, la *top-1 accuracy*, la *pr@10* et la *pr@100*. Afin de déterminer quelle stratégie d'augmentation est la plus efficace, nous classons de la meilleure à la moins bonne chaque expérience. Cela nous permet ainsi de calculer la moyenne des rangs qu'a obtenu chaque stratégie d'augmentation et donc d'avoir un score unifié de leurs performances.

Un grand nombre d'expérimentations ont été réalisées (combinaison de dégradations). Les résultats les plus significatifs de ces expériences sont visibles sur la Figure. 4.6 [4].

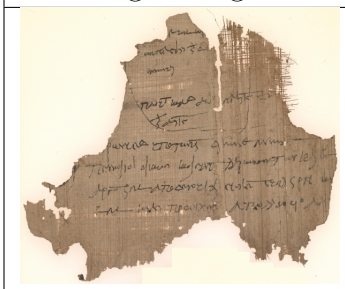
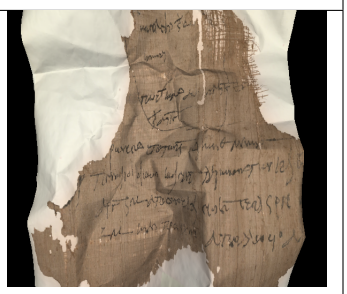
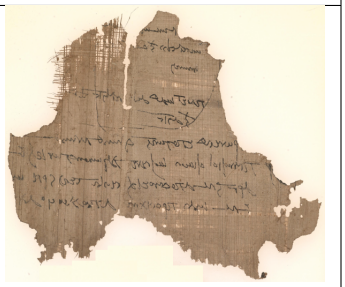
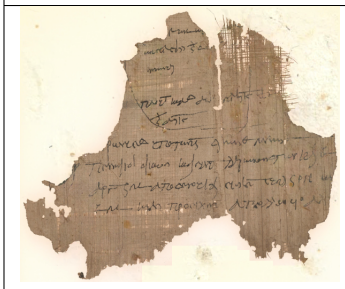
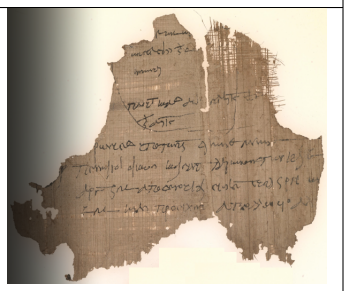
| Image d'origine | Déformation 3D |
|---|---|
|  |  |
| Ajout de trous | miroir horizontal |
|  |  |
| Ajout de taches | Défaut d'illumination |
|  |  |

Tableau 4.7 – Exemple d'images générées avec DocCreator. Ces images sont utilisées pour déterminer quel algorithme (ou combinaison d'algorithmes) est utile au processus d'apprentissage.

4. Ces résultats ont été produits dans le cadre du stage de *DUT* de Priscilla Tissot.

D'après ces expériences préliminaires, tous les types d'augmentation de données ne contribuent pas à améliorer les résultats. On observe principalement qu'effectuer des transformations 3D qui rajoutent par exemple des plis aux papyrus, des transformations 2D élastiques, ou encore rajouter du flou dégrade significativement les performances. Au contraire, on observe qu'effectuer un miroir horizontal (la ligne *BaseFlipH* sur la Figure. 4.6) améliore par exemple significativement les performances.

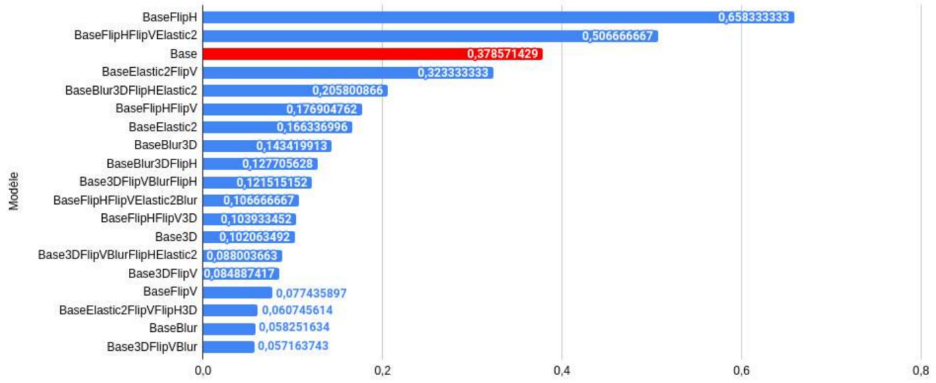



FIGURE 4.6 – Comparaison de différentes combinaisons d'augmentations de données. En rouge, le résultat obtenu sans augmentation de données.

Dans cette expérience, les documents réels utilisés pour générer des versions synthétiques l'ont été de manière aléatoire (nombre d'images, paramètres des algorithmes de déformation). En nous basant sur ces résultats, nous étudions dans la section suivante la possibilité de tirer encore plus profit de données synthétiques en contrôlant la manière dont elles sont utilisées dans la constitution des batchs.

### 4.4.2 Sélection dans les batchs

Une autre technique qui a montré son intérêt pour améliorer la convergence des modèles est ce qu'on appellera la "sélection dans les batchs". Il s'agit de sélectionner dynamiquement lors de l'entraînement les exemples qui vont être les plus "informatifs" à une étape donnée de l'entraînement. Certains exemples sont en effet plus difficiles que d'autres à traiter. Il est efficace au début de l'entraînement de limiter la difficulté des exemples qui sont donnés au modèle pour accélérer son apprentissage (Gao et Jojic (2016)). À l'inverse, lorsque l'entraînement est plus avancé, certains exemples deviennent "faciles" pour le modèle, c'est-à-dire qu'il est capable de les classifier correctement avec une grande confiance. À ce moment, il est efficace de réduire le nombre de ces exemples faciles qui sont présentés au modèle, car ils ne contribuent quasiment plus à la progression de l'entraînement et ralentissent donc la convergence (Shrivastava et al. (2016)).

Certains jeux de données sont plus difficiles que d'autres en eux mêmes, c'est un facteur important de la stratégie de sélection à adopter. Quand la tâche est difficile (base difficile ou bruitée) on obtient de meilleures performances en insistant sur les exemples faciles (Chang et al. (2017)). Il semble cependant contreproductif de construire des batchs de difficulté homogène d'après (Gao et Jojic (2016)).

On peut généraliser l'approche du "minage de triplets en ligne" introduite par Schroff et al. (2015) à n'importe quel type de données présentes dans les batchs. Pour rappel, l'idée est qu'en formant tous les triplets d'exemples d'entraînement possibles, certains seront plus faciles que d'autres. Dans le cas de la *Triplet Loss*, elle retournera même 0 dans le cas d'un triplet "facile", ce qui fait que ce triplet ne contribue pas à la progression de l'entraînement et ralentit donc la convergence (cf. Section 2.22 pour des explications).

Concrètement, à chaque étape de l'entraînement, on peut connaître parmi les exemples que l'on donne au modèle dans un batch lesquels sont "faciles" (loss faible) et lesquels sont "difficiles" (loss élevée). Les exemples "faciles" contribuent moins à la convergence que les exemples difficiles (Shrivastava et al. (2016)). Il peut donc être intéressant de ne sélectionner que des exemples difficiles. On réduit ainsi la taille du batch, on accélère l'entraînement. On peut définir dans ce cas comme "faciles" les exemples qui sont correctement classifiés avec une marge définie (Shrivastava et al. (2016))

### 4.4.3 Combiner augmentation de données et sélection dans les batchs

D'après les différentes recommandations qui émergent de la littérature, dont nous avons parlé juste avant, et les conclusions de l'étude sur l'impact de différents types d'augmentation de données, nous mettons en place le protocole expérimental suivant :

Pour des batch de taille 64:
— Déclenchement du processus de sélection dans les batchs à l'epoch 10.
— Duplication des 8 paires les plus difficiles.
— Les 8 paires les plus faciles sont remplacées par les copies des 8 paires les plus difficiles qui ont été transformées avec l'algorithme d'augmentation de données qui a donné le meilleur résultat (*hflip*).

Dans les expériences reportées dans le Tableau. 4.8, les modèles ont été entraînés sur une base de *VGG16* en mode *auto-supervisé* sur la base *Michigan*, avec 5 patchs par fragments, 2195 fragments d'entraînement et 258 fragments de test (soit 58 documents). Les chiffres ne sont ainsi pas directement comparables avec ceux des chapitres précédents, qui utilisaient des quantités de données de test différentes. Nous avons néanmoins recalculé une *Baseline* permettant d'évaluer l'impact relatif

des différentes stratégies d'augmentation et de sélection. Cette *Baseline* est entraînée en mode *auto-supervisé* de la même manière que dans la Section. 4.2.2.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| Baseline | 0.44 | 0.68 | 0.52 | 0.91 |
| Augment_*hflip* | 0.45 | 0.67 | 0.54 | 0.93 |
| Trigger_10 | 0.48 | 0.77 | 0.55 | 0.91 |
| Trigger_10_augment_*hflip* | **0.50** | **0.80** | **0.57** | **0.91** |
| Trigger_10_augment_*hflip*_*vflip* | 0.48 | 0.77 | 0.56 | 0.92 |

TABLEAU 4.8 – Comparaison des stratégies de sélection dans les batchs avec de l'augmentation de données. *Trigger_10* signifie que le processus de sélection dans les batchs a été déclenché à partir de l'epoch 10.

Nous pouvons voir dans le Tableau. 4.8 que la stratégie combinant sélection dans les batchs et augmentation de données par *hflip* améliore les performances de 6 points par rapport à la *Baseline*. On remarque aussi que le fait de seulement retirer les exemples les plus faciles et de les remplacer par les exemples les plus difficiles dupliqués améliore de 4 points les performances par rapport à la *Baseline*. Il est aussi intéressant de noter qu'en introduisant un miroir vertical (*vflip*) en plus du *hflip* et de la sélection dans les batchs, on perd 2 points par rapport à la stratégie n'utilisant que le *hflip* et la sélection (dernière ligne du Tableau. 4.8).

Ces expériences sont une première exploration pour déterminer si ces stratégies ont le potentiel d'améliorer les performances de notre approche auto-supervisée. Plus d'expériences sont nécessaires pour tirer des conclusions significatives, mais il semble au premier abord que cette approche combinant sélection dans les batchs et augmentation de données, ne nécessitant par ailleurs pas de travail humain supplémentaire, pourrait être intéressante pour gagner en performance.

## Conclusion

Nous avons montré qu'il est possible d'obtenir des résultats exploitables en réduisant le plus possible la quantité de travail manuel d'annotations par les experts. L'approche de l'apprentissage par transfert avec ré-entraînement sur une petite quantité de documents annotés a montré son intérêt. Nous concluons cependant qu'il est encore plus intéressant de partir d'un modèle pré-entraîné sur une base du même domaine et de ré-entraîner en mode *auto-supervisé* avec toutes les données non annotées de la base "objectif» disponibles. Dans ce cas précis, l'utilisation de données annotées pendant le ré-entraînement n'apporte qu'un gain marginal (cf. 4.2).

Nous avons aussi montré qu'il est possible de faire des prédictions sur des petits corpus de documents à partir seulement de ces corpus eux-mêmes sans avoir besoin

d'effectuer des annotations. Étant donnés les résultats obtenus sur ces deux bases très différentes en termes de types de documents, de nombre de fragments par document et de tailles de fragments, nous sommes confiants dans la capacité de cette approche à se transposer sur d'autres jeux de données.

Enfin, nous avons voulu commencer à explorer une stratégie combinant sélection d'exemples difficiles dans les batchs avec augmentation de données. Ces premiers résultats nous incitent à penser que cette approche a le potentiel d'améliorer les performances de nos modèles, mais plus de travail est nécessaire pour quantifier l'intérêt d'une telle stratégie.

Dans le prochain chapitre, nous présentons une application de notre approche à un cas réel : la collection *Jouguet* du projet *GESHAEM*. Nous y étudions les propositions d'assemblage proposées par un modèle entraîné en mode *auto-supervisé* sur les données de la collection *Jouguet*.

# Chapitre 5

# Application à *GESHAEM*

## Sommaire

# Introduction

Le projet *GESHAEM*, dans lequel s'inscrivent ces travaux de thèse, a pour objectif de mieux comprendre l'administration d'une région agricole d'Égypte sous domination grecque pendant le 3ème siècle AEC. C'est au travers de l'étude des papyrus de la collection *Jouguet* que les chercheurs du projet *GESHAEM* souhaitent approfondir les grands changements qui ont eu lieu à cette époque, en termes de fonctionnement de l'administration et des processus agricoles. Des détails sur le projet *GESHAEM* et sur la collection *Jouguet* sont donnés dans la Section. 1.2.

Ce dernier chapitre propose de détailler les tests réalisés sur le fond *Jouguet*. Les fragments composant cette collection sont plus complexes à étudier que ceux sur lesquels notre méthode a été évaluée dans les chapitres précédents. En effet, nous disposons de peu de fragments, certains sont particulièrement petits, contiennent parfois peu de lignes de texte et peuvent être très dégradés.

Les tests sur la collection *Jouguet* sont complexes à réaliser, car nous disposons de très peu de vérité terrain. Depuis le début du projet, une cinquantaine de raccords corrects ont été trouvés manuellement par les membres du projet *GESHAEM*. Ces raccords ont été faits en étudiant le texte, l'épaisseur des fibres et la manière dont étaient collés certains fragments (kollesis). Cette petite quantité d'images annotées ne nous a pas empêchés de mener une étude qualitative et quantitative. Cette analyse a été faite en collaboration avec Marie-Pierre Chaufray, la papyrologue en charge du projet *GESHAEM*.

Un autre enjeu lié à l'application de nos recherches a été le développement d'une plateforme logicielle permettant de manipuler les fragments de papyrus numérisés, et d'accéder aux suggestions d'appariements proposées par notre modèle. Le logiciel *Orio* a été développé pour répondre à ces besoins.

Ce chapitre est organisé de la manière suivante : la Section. 5.1 détaille la collection *Jouguet* et la manière dont elle a été numérisée, la Section. 5.2 présente le logiciel *Orio* et enfin, la section 5.3 présente les résultats obtenus sur la collection *Jouguet* en utilisant notre méthode d'apprentissage *auto-supervisée*, combinée à un pré-entraînement et de l'augmentation de données conjointement à de la sélection dans les batchs.

## 5.1 La collection *Jouguet*

### 5.1.1 Les fragments de papyrus de la collection *Jouguet*

Comme expliqué dans la Section. 1.2, les fragments de la collection *Jouguet* ont été extraits de cartonnages de momies. Ces ornements funéraires étaient fabriqués à

partir de papyrus réutilisés, qui étaient déchirés et collés pour créer une forme, puis peints. Au début du XXe siècle, *Pierre Jouguet* effectue des fouilles dans la région du Fayoum en Égypte et retrouve de tels cartonnages de momies. Il procède alors à l'extraction et à l'inventorisation des fragments provenant de ces cartonnages. Le numéro d'inventaire de chaque fragment est constitué d'une partie numérique et d'une lettre. Des fragments partageant la même partie numérique proviennent du même cartonnage. Par exemple, le fragment **1237a** provient du même cartonnage que le fragment **1237b**. Il pourra aussi exister les fragments **1237c**, **1237d**, etc.

Le processus original de construction des cartonnages, le passage du temps et le processus d'extraction des fragments ont contribué à l'aspect dégradé qu'ont aujourd'hui les fragments de papyrus de la collection *Jouguet*, comme on peut le voir sur la Figure. 5.1. On voit bien par exemple sur le premier fragment à gauche de la Figure. 5.1 la structure du support d'écriture qui était confectionné à l'époque : des fibres horizontales sont superposées et collées orthogonalement à des fibres verticales. En déchirant ce papyrus, les fibres horizontales se sont décollées et on peut ainsi voir que la ligne d'écriture tout en haut est coupée. On peut aussi voir que l'écriture est parfois difficilement lisible, car en partie effacée. Ces facteurs rendent la reconstruction des papyrus difficile.

FIGURE 5.1 – Exemples de fragments de la collection *Jouguet*

Les papyrologues se basent sur plusieurs critères pour déterminer si deux fragments peuvent être raccordés. Lorsque le texte est lisible, il est parfois possible de trouver la suite d'une ligne sur un autre fragment, ou alors de déterminer que le propos du texte est similaire entre les fragments. La ressemblance du style d'écriture et de l'épaisseur de l'encre peut aussi être un critère fort pour appairer des fragments. Parfois, deux fragments peuvent être raccordés grâce à des spécificités dans l'aspect des fibres qui peuvent se continuer d'un fragment à l'autre. Il arrive aussi que deux fragments soient appairés grâce à la présence de *Kollesis*[1] qui correspondent sur les deux fragments. Enfin, la structure du document est importante pour déterminer si un appairage est possible entre deux candidats. Les cautionnements ont par exemple la caractéristique d'être écrits sous la forme d'une colonne étroite de texte. Une autre

1. Un raccord entre deux feuilles de papyrus qui sont collées l'une à l'autre

hypothèse avec laquelle travaillent les papyrologues est qu'il est plus probable que deux fragments provenant du même cartonnage puissent être raccordés que deux fragments provenant de cartonnages différents. Cette hypothèse nous permettra par la suite d'étudier de possibles corrélations entre les propositions d'assemblages données par notre modèle et l'appartenance des fragments à un cartonnage commun.

### 5.1.2 Acquisition numérique des fragments de papyrus

Au début du projet a été mis en place un protocole d'acquisition photographique des fragments de papyrus de la collection *Jouguet*. Il a été décidé de photographier les fragments à la fois en couleur et avec un filtre infrarouge.


FIGURE 5.2 – Exemple d'image issue de la base *GESHAEM*. Photographie couleur à gauche et photographie infrarouge à droite

Cela permet de faire ressortir les pigments de l'encre comme on peut le voir sur la Figure. 5.2, car ces pigments absorbent presque toute la lumière dans le spectre infrarouge, alors que le papyrus la réfléchit en grande partie. Cela crée un contraste important qui facilite la lecture. De plus, il est possible que des pigments qui ont été dégradés avec le temps n'aient plus de couleur "visible", c'est-à-dire qu'ils n'absorbent plus les longueurs d'onde du spectre visible par l'œil humain, mais absorbent toujours d'autres longueurs d'onde, notamment les infrarouges. Cela peut permettre de faire apparaître des écritures qui sont effacées.

En préparation de l'acquisition, chaque fragment est placé sur une feuille blanche et trois éléments sont positionnés à côté de lui (cf. Fig. 5.2). Son numéro d'inventaire,

FIGURE 5.3 – Le dispositif d'acquisition couleur et infrarouge des papyrus. Sur la photo apparaissent des membres du projet *GESHAEM* : Adam Bülow-Jacobsen (au premier plan), Marie-Pierre Chaufray (à gauche) et Lorenzo Uggetti

écrit sur un morceau de papier, une règle graduée, qui permet aux papyrologues de connaître la taille réelle du fragment lorsqu'ils le consultent numériquement et une équerre en papier sur laquelle sont imprimés des *Tags ArUco* Romero-Ramirez et al. (2018). Cette équerre a pour but de faciliter l'automatisation de l'extraction des dimensions du fragment photographié. La bibliothèque *ArUco*[2] permet en effet de facilement détecter ces marqueurs, et puisque l'espacement entre les marqueurs est connu et ne change jamais, on peut aisément automatiser le calcul de la taille réelle du fragment photographié.

Puisque les images couleur et infrarouges sont acquises avec deux appareils photo indépendants placés sur deux trépieds (cf. Fig. 5.3), les deux images présentent des différences au niveau de la disposition des éléments. En effet, la plaque contenant le fragment, la règle, l'équerre et le numéro d'inventaire est déplacée d'un appareil photo à l'autre manuellement et le cadrage n'est pas parfaitement identique entre les deux photos. Ainsi, les deux images ne sont pas superposables telles quelles et

2. http://www.uco.es/investiga/grupos/ava/node/26

un traitement est nécessaire si l'on souhaite les re-aligner. Tout ce processus est effectué pour le recto et le verso du fragment. Tout cela est traité en détail dans Célor (2018).

Au moment de la rédaction de ce manuscrit, 381 fragments ont été numérisés et indexés sur une base de données en ligne. Pour chaque fragment, nous avons quatre photographies (recto/verso et couleur/infrarouge).

## 5.2 Développement d'un logiciel de manipulation de fragments de papyrus : *Orio*

Ce travail de thèse a été réalisé en collaboration avec le laboratoire Ausonius. Nous avons donc eu l'opportunité d'échanger avec des chercheurs qui ont l'habitude de réaliser manuellement le travail d'association des fragments. Il est apparu que cette tâche de manipulation d'images de fragments numérisés est commune à plusieurs laboratoires de recherche en papyrologie dans le monde. C'est dans cette optique que le logiciel **Orio**[3] a été développé. Il vise à faciliter la manipulation de fragments de papyrus numériques par les papyrologues, et de leur permettre d'accéder facilement aux suggestions d'appariements proposées par notre modèle. Il vise à simuler la façon dont peuvent travailler des papyrologues lorsqu'ils cherchent à raccorder des fragments physiques. Il a été développé en javascript avec la bibliothèque *React*[4] et le *framework Electron*[5]. Cela permet à l'application d'être directement exportable en exécutable compatible *Linux, Windows et MacOS*.

Comme l'illustre la Figure. 5.4, **Orio** est pensé pour s'interfacer avec un serveur web *Nakala*[6] pour aller récupérer les images de papyrus qui y sont stockées. *Nakala* est un entrepôt de données spécialisé dans le stockage d'images, de textes ou de vidéos issus de travaux de recherche en humanités numériques. Cela facilite la mise à jour des données au fur et à mesure que de nouvelles séances d'acquisition ont lieu, il suffit de mettre à jour le serveur *Nakala* pour que tous les utilisateurs d'**Orio** travaillent directement sur les données les plus à jour. Une collection de scripts permettant de segmenter les nouvelles images et de calculer les nouvelles suggestions d'appairage grâce à notre modèle est mise à disposition de la personne gérant le serveur.

On peut voir une capture d'écran de l'interface graphique du logiciel sur la Figure. 5.5. La zone **1** est une liste déroulante des données disponibles. Il est possible de

3. **Orio** a été développé durant différents stages étudiants au sein du projet *GESHAEM*. J'ai co-encadré les étudiants et contribué au logiciel, dont le code source peut être trouvé ici : https://gitlab.huma-num.fr/orio/orio
4. https://fr.reactjs.org/
5. https://www.electronjs.org/
6. https://nakala.fr

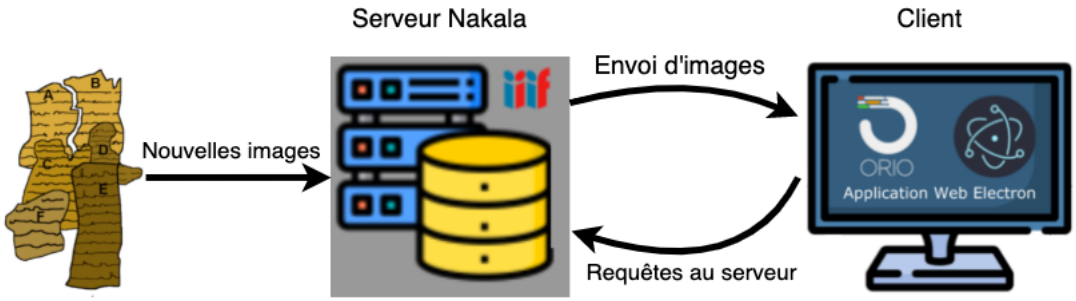



FIGURE 5.4 – Architecture du logiciel **Orio**. Orio est un logiciel libre de droits qui permet à des experts en papyrologie de se connecter à un serveur Nakala, de télécharger des images et demander des suggestions d'appairage de fragments.

prévisualiser le fragment en plus grand en le survolant, une fenêtre apparaît alors (le point **3**) pour mieux visualiser le fragment. Lorsqu'un fragment est glissé depuis la zone **1** dans la zone **2**, une version segmentée de ce fragment est ajoutée à l'espace de travail. On peut alors visualiser en simultané le recto et le verso dans l'espace de travail *recto* (partie de gauche) et l'espace de travail *verso* (partie de droite). Si une des deux images (recto ou verso) est manquante pour un fragment, elle est remplacée par l'autre image, en miroir horizontal et avec une opacité différente pour l'indiquer, comme on peut le voir sur la Figure. 5.5 pour le fragment en haut à gauche de l'espace de travail *verso*. Il est ensuite possible de déplacer les fragments et de leur appliquer des rotations. Ces modifications sont synchronisées entre l'espace de travail *recto* et l'espace de travail *verso*.

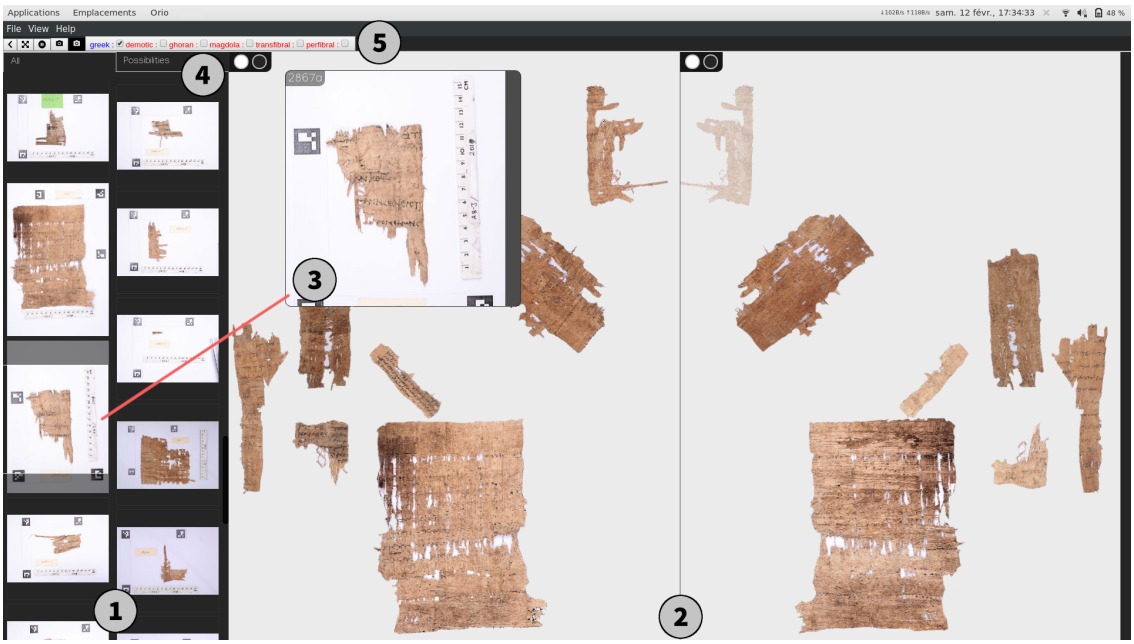



FIGURE 5.5 – L'interface graphique d'**Orio**

L'utilisateur peut effectuer un clic droit sur un fragment qui est dans l'espace de travail et demander des suggestions d'appairages pour ce fragment. La liste ordonnée des fragments les plus similaires d'après notre modèle est alors affichée dans l'onglet "*Possibilities*" (au niveau du point **4**). Enfin, l'utilisateur a la possibilité de filtrer les listes de fragments avec différents critères (point **5**). Ici, il est par exemple possible de filtrer par langue (grec ou démotique), par localisation géographique (Ghoran ou Magdola) ou par le sens de la fibre par rapport à l'écriture (transfibral ou perfibral).

## 5.3 Analyse des résultats sur la collection *Jouguet*

### 5.3.1 Entraînement du modèle

D'après les conclusions tirées des chapitres 3 et 4, nous avons mis en place un certain nombre de modalités d'apprentissage pour entraîner un modèle fonctionnant sur les données de la collection *Jouguet*.

Premièrement, ne disposant que d'extrêmement peu de données annotées, nous exploitons l'approche *auto-supervisée* décrite dans la Section. 4.2. Nous avons aussi vu dans la Section. 4.2.2 qu'il était préférable de partir d'un modèle pré-entraîné. Nous utiliserons donc le modèle entraîné sur *hisfrag* ayant donné les meilleurs résultats après ré-entraînement en mode *auto-supervisé* sur la base *Michigan*, à savoir le modèle basé sur *Resnet50* qui a obtenu 0.43 de *mAP* (cf. Tableau. 4.6).

Comme pour les expériences précédentes, on construit des masques des papyrus et on extrait les sous-lignes à l'aide de *ARU-Net* (cf. Section. 3.3.3). Grâce à ces deux informations, on extrait des patchs qui contiennent un maximum de texte et ne sont pas positionnés sur la bordure du papyrus (cf. Section. 3.4.1).

Dans le modèle siamois décrit dans la Section. 3.2, nous utilisons la partie conventionnelle de l'architecture *Resnet50* et les batchs sont équilibrés à l'entraînement (cf. Section. 3.2.3). Le modèle est entraîné avec des patchs de taille 64, une taille de batchs de 128 pendant 100 epochs. Nous rajoutons enfin la procédure de sélection dans les batchs et d'augmentation de données, telle que décrite dans la Section. 4.4. Toutes les images de la collection *Jouguet* sont utilisées pour l'entraînement et la validation (avec une séparation de 80% pour l'entraînement et 20% pour la validation).

Une fois le modèle entraîné, nous calculons une matrice de similarité entre tous les fragments. Chaque fragment est comparé à tous les autres fragments de la base. On peut ensuite pour chaque fragment, fournir une liste classée des autres fragments, du plus similaire au moins similaire. Par exemple, sur la Figure. 5.6 on peut voir les 5 fragments qui ont été les mieux classés par rapport au fragment requête (tout à gauche).


FIGURE 5.6 – Exemple de requête. Le fragment requête est celui tout à gauche. À sa droite, les 5 fragments ayant les plus grands scores de similarité d'après le modèle.

### 5.3.2 Étude des suggestions proposées par le modèle sur les papyrus de la collection *Jouguet*

Un petit ensemble de fragments ont déjà été raccordés avec certitude par les papyrologues. Nous comparons dans cette section les prédictions de notre modèle avec les raccords existants. Tous les classements calculés par notre modèle sur les raccords fournis par les papyrologues peuvent être consultés sur le Tableau. 5.2 de l'Annexe. B. Certains fragments ont été raccordés avec plusieurs autres fragments (les fragments **2855a**, **2855c**, **2855d** et **2855e** par exemple).

**Note :** on peut remarquer sur le Tableau. 5.2 que le classement calculé n'est pas nécessairement le même pour deux fragments donnés en fonction duquel était le fragment requête ($classement(A \rightarrow B) \neq classement(B \rightarrow A)$). Cela est simplement dû au fait que d'autres fragments ont pu être mieux classés par rapport au fragment **A** que le fragment **B**. Le score de similarité entre les deux fragments est cependant le même, quel que soit l'ordre.

| Classement | Nombre |
|---|---|
| Première place | 6 |
| Seconde place | 2 |
| Troisième place | 0 |
| Top 10 | 12 |
| Top 20 | 12 |
| Top 50 | 19 |
| Top 100 | 24 |
| Total | 42 |

TABLEAU 5.1 – Synthèse des classements retournés par notre modèle pour les fragments dont on connaît les raccords existants.

Une synthèse de ces résultats peut aussi être consultée sur le Tableau. 5.1. On peut par exemple y voir que sur les 42 paires de fragments raccordées par les papy-

rologues, nous en avons classé 6 en première position, 2 en deuxième position et 12 dans le top 10. Il est cependant à noter que quasiment la moitié des paires ont été classées très loin, après la 100ème position. Ces mauvais résultats peuvent parfois s'expliquer en analysant plus précisément les fragments en question, en effet, certains fragments sont parfois très petits et ne contiennent donc que très peu d'information exploitable par le modèle.

**Examinons quelques cas qui ont bien fonctionné.**

### Les fragments 1317b et 1378c

Les fragments **1317b** et **1378c** ont par exemple été raccordés par les papyrologues, comme on peut le voir sur la Figure. 5.7 [7]. Dans la liste classée de fragments suggérés comme étant potentiellement similaires au fragment **1378c** produite par notre modèle, le fragment **1317b** a été classé à la quatrième position. En première et deuxième position, on retrouve deux fragments provenant du même cartonnage et en troisième position le fragment **1311**.


FIGURE 5.7 – Raccord physique (à droite) entre le fragment **1378c** (à gauche) et le fragment **1317b** (au milieu). © Sorbonne Université - Institut de Papyrologie.

### Les fragments 0750b et 0796

Le modèle a aussi su prédire avec succès certains cas jugés très difficiles par les papyrologues. Par exemple, le fragment **0750b** a été classé en première position avec le fragment **0796** avec un score de similarité de 0.87 alors que ce premier est un petit fragment ne contenant qu'une ligne de texte rognée (cf. Fig. 5.8).

7. Les raccords physiques présents sur la droite de la Figure. 5.7, en haut de la Figure. 5.10 et sur la Figure. 5.11 ont été réalisés par Florent Jacques, Papyrothécaire à l'Institut de la Papyrologie de la Sorbonne et membre du projet **GESHAEM**.

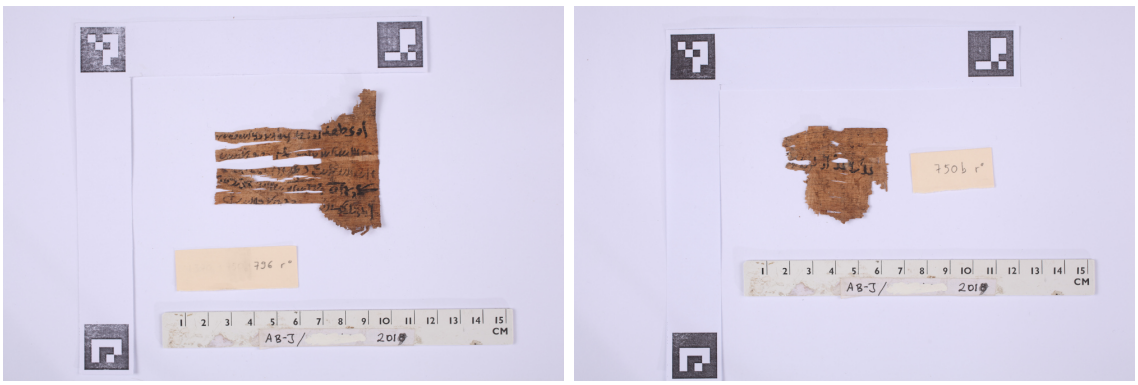


FIGURE 5.8 – Les fragments **0796** (à gauche) et **0750b** (à droite).

### Les fragments 0812a, 0812b et 2275

D'après la liste de raccords fournis par les papyrologues, les fragments **0812a** et **0812b** sont associés au fragment **2275** (cf. Fig. 5.9). Notre modèle a correctement classé ces deux fragments en première et deuxième position par rapport au fragment **2275**.

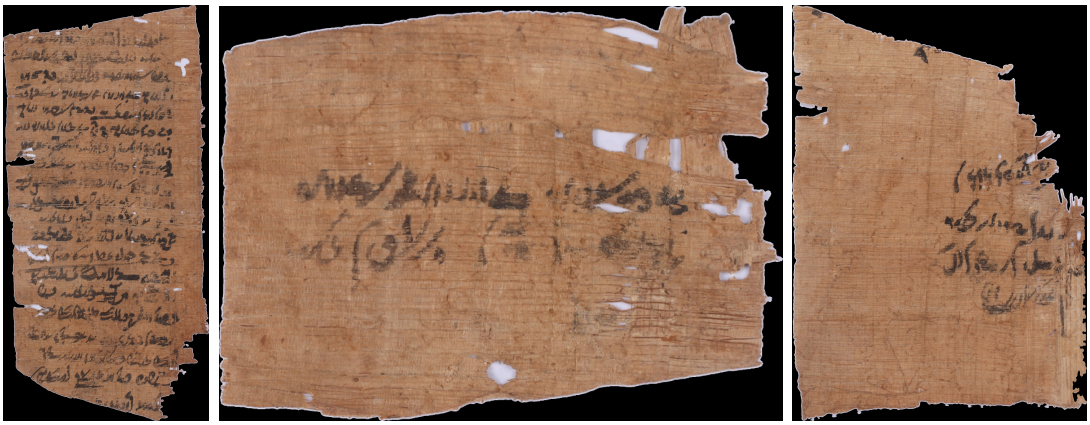


FIGURE 5.9 – De gauche à droite, les fragments **2275**, **0812a** et **0812b**

### Les fragments 1382d et 1382e

De même, les fragments **1382d** et **1382e** ont été raccordés par les papyrologues (cf. Fig. 5.10), et notre modèle a classé le fragment **1382e** en deuxième position par rapport au fragment **1382d** avec un score de similarité de 0.74.

### Les fragments 2733, 802 et 803b

Un autre raccord qui a été relativement bien prédit par notre modèle est celui entre les fragments **2733**, **802** et **803b**. Le fragment **802** a été classé en 12ème

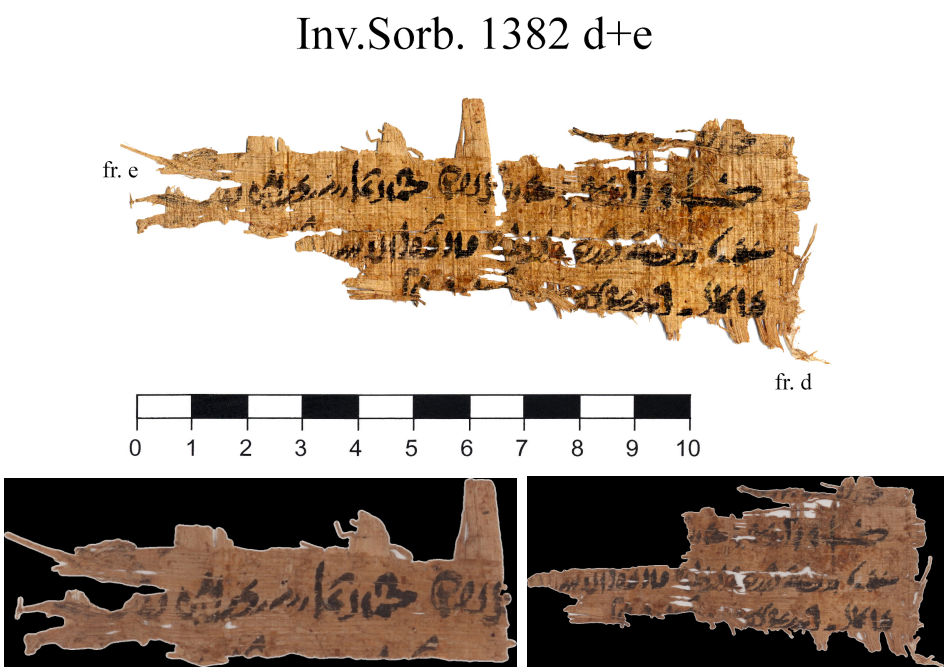



FIGURE 5.10 – Raccord physique (en haut) entre le fragment **1382d** (à gauche) et le fragment **1382e** (à droite). © Sorbonne Université - Institut de Papyrologie.

position par rapport au fragment **2733**, et le fragment **803b** en 14ème position. Le fragment **1265b** est aussi raccordé à tous ces fragments (cf. Fig. 5.11), mais il ne faisait pas partie de notre jeu de données lors du calcul des similarités, il a été photographié plus tard.

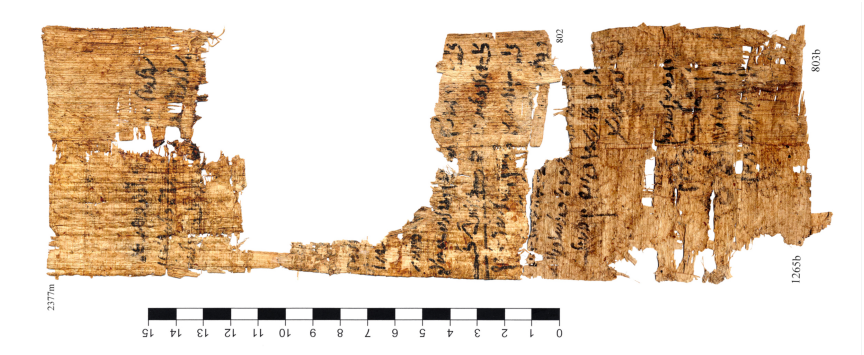


FIGURE 5.11 – Raccord physique entre les fragments **2733m**, **802**, **803b** et **1265b**. © Sorbonne Université - Institut de Papyrologie.

**Examinons quelques cas qui n'ont pas bien fonctionné.**

### Les fragments 2855e, 2855d et 2855a

Le fragment **2855e** a été classé à la 298ème position (sur 381) par rapport au fragment **2855d**, ce qui est un résultat très mauvais pour deux fragments qui ont été raccordés par les papyrologues. Cependant, en regardant les fragments, on remarque que le fragment **2855e** est minuscule (cf. Fig. 5.12) et ne contient donc que très peu d'information pouvant être utile à notre modèle. Les papyrologues ont raccordé ces deux fragments par élimination, car ils proviennent du même cartonnage et que l'écriture est similaire.

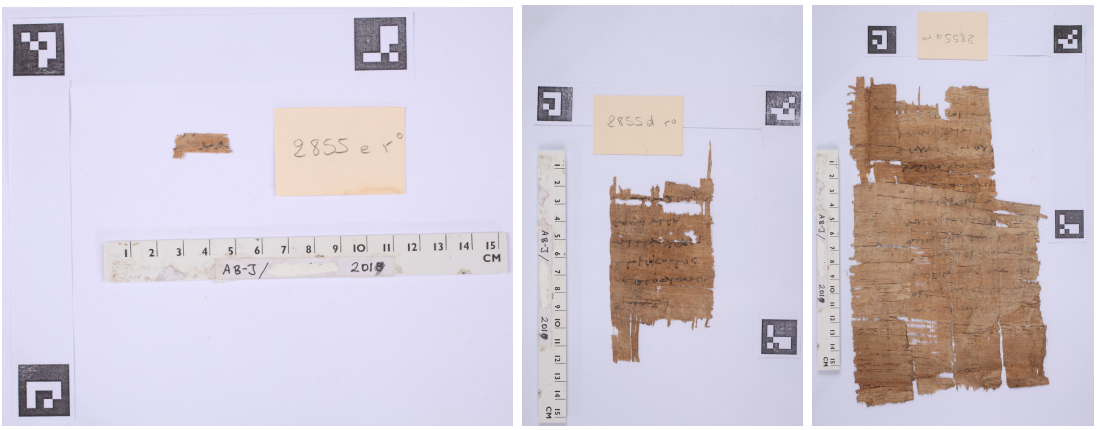


FIGURE 5.12 – Les fragments **2855e** (en haut), **2855d** (à gauche) et **2855a** (à droite)

Le fragment **2855a** a quant à lui été classé à la 50ème position par rapport au fragment **2855d**. Ce classement médiocre est dans ce cas un peu plus étonnant, car le fragment est de taille raisonnable et le style du texte est très similaire d'après les papyrologues. On peut cependant voir que l'écriture est beaucoup plus effacée sur le fragment **2855a** que sur le fragment **2855d** (cf. Fig. 5.12).

### Les fragments 1419 et 0567h

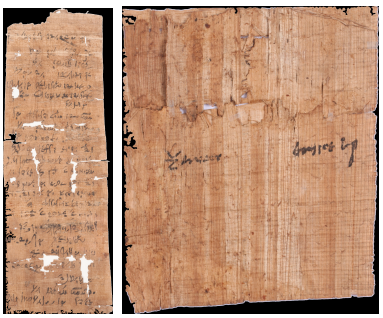


FIGURE 5.13 – Les fragments **1419** (à gauche) et **0567h** (à droite)

Un autre cas intéressant est celui des fragments **1419** et **0567h**. Le fragment **0567h** a été classé en première position par rapport au fragment **1419** avec un score de similarité de 0.94. D'après la papyrologue du projet Marie-Pierre Chaufray, cette association est improbable, car le fragment **0567h** est un *cautionnement* alors que le fragment **1419** est un *compte*. De plus, le fragment **1419** est probablement entier en hauteur, ce qui diminue encore la probabilité de cette association (cf. Fig. 5.13).

### Les fragments 1438g et 1438p

Enfin, le fragment **1438g** a été classé en première position avec le fragment **1438p** avec un score de similarité de 0.95 (cf. Fig. 5.14). Encore une fois, ces deux fragments ne peuvent pas provenir du même papyrus d'après Marie-Pierre Chaufray car ils contiennent tous les deux le début d'un contrat, mais ils peuvent cependant avoir été écrits par le même scribe. En effet, ils proviennent du même cartonnage, et ils ont été écrits à la même date (cette information est contenue dans le texte).

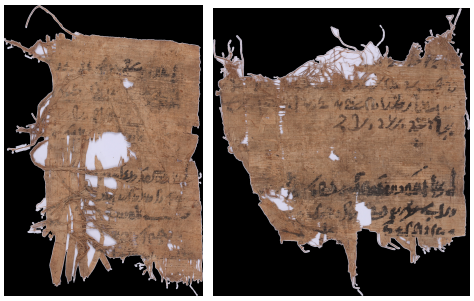


FIGURE 5.14 – Les fragments **1438g** (à gauche) et **1438p** (à droite)

### Les fragments 2733f et 0803a

Un autre exemple de ce type est le fragment **2733f** qui est classé premier par rapport au fragment **0803a** avec un score de similarité de 0.97 et qui a vraisemblablement été écrit par le même scribe (cf. Fig. 5.15).

FIGURE 5.15 – Les fragments **0803a** (verso, à gauche) et **2733f** (à droite).

### 5.3.3 Étude sur les cartonnages

Comme nous l'avons dit plus tôt, les papyrologues travaillent avec l'hypothèse que des fragments dont le numéro d'inventaire partage la partie numérique, ou dont cette partie numérique est voisine sont plus susceptibles de provenir du même document que des fragments ayant des numéros éloignés (ex : les fragments **1234a** et **1234b** proviennent du même cartonnage). En effet, la plupart du temps la partie numérique du numéro d'inventaire correspond à un cartonnage unique. Il existe cependant des exceptions dans lesquelles des fragments ont été extraits du même cartonnage, mais la partie numérique de leur numéro d'inventaire n'est pas la même. Cela provient de la méthode d'inventorisation qui a été appliquée lors des fouilles de *Pierre Jouguet* au début du XXe siècle, dont certains détails sont encore méconnus aujourd'hui. De manière générale, on peut considérer que des numéros d'inventaires proches correspondent probablement à des fragments issus de cartonnages qui ont été retrouvés proches géographiquement (sur la même momie par exemple).

Cette hypothèse s'explique par le procédé de fabrication des cartonnages. Lors de leurs créations en Égypte il y a plus de 2000 ans, les artisans utilisaient des papyrus dont le contenu était alors obsolète pour fabriquer ces ornements funéraires. Ils déchiraient ou découpaient ces papyrus pour qu'ils aient la forme dont ils avaient besoin. Ainsi, on peut imaginer que les différents fragments d'un papyrus donné soient utilisés pour la confection d'un cartonnage donné. Il est en somme improbable qu'ils aient au préalable déchiré et mélangé tous les papyrus à leurs dispositions, puis puisé dans cette collection de fragments mélangés. Ainsi, si cette hypothèse se vérifie et que notre modèle est efficace, nous devrions remarquer une corrélation entre scores de similarité élevés et proximité des numéros d'inventaire.

On peut observer ce phénomène sur la Figure. 5.16. Cette matrice permet de visualiser les scores de similarité calculés sur les 381 fragments de la collection *Jouguet* auxquels nous avons accès. En ligne et en colonne, les noms des fragments sont triés dans l'ordre alphabétique, ce qui a pour effet de mettre côte à côte les fragments dont la partie numérique est la même ou voisine. Ainsi, sur la diagonale est représenté le score de similarité de chaque fragment avec lui-même (qui est donc égal à 1).

Le fait d'avoir classé les numéros d'inventaire par ordre alphabétique permet de visualiser les scores des fragments qui appartiennent au même cartonnage, ou qui proviennent de cartonnages voisins. On peut le voir sous la forme de "blocs" de pixels jaunes (qui correspond à un score proche de 1, cf. échelle de couleur à droite de la Figure. 5.16). Si un de ces "blocs" apparaît au niveau de la diagonale, cela signifie que des fragments provenant du même cartonnage, ou de cartonnages de numéros voisins ont été identifiés comme "similaires" par notre modèle. Un "bloc" qui apparaît ailleurs que sur la diagonale indique potentiellement un lien entre deux cartonnages éloignés identifiés par notre modèle.

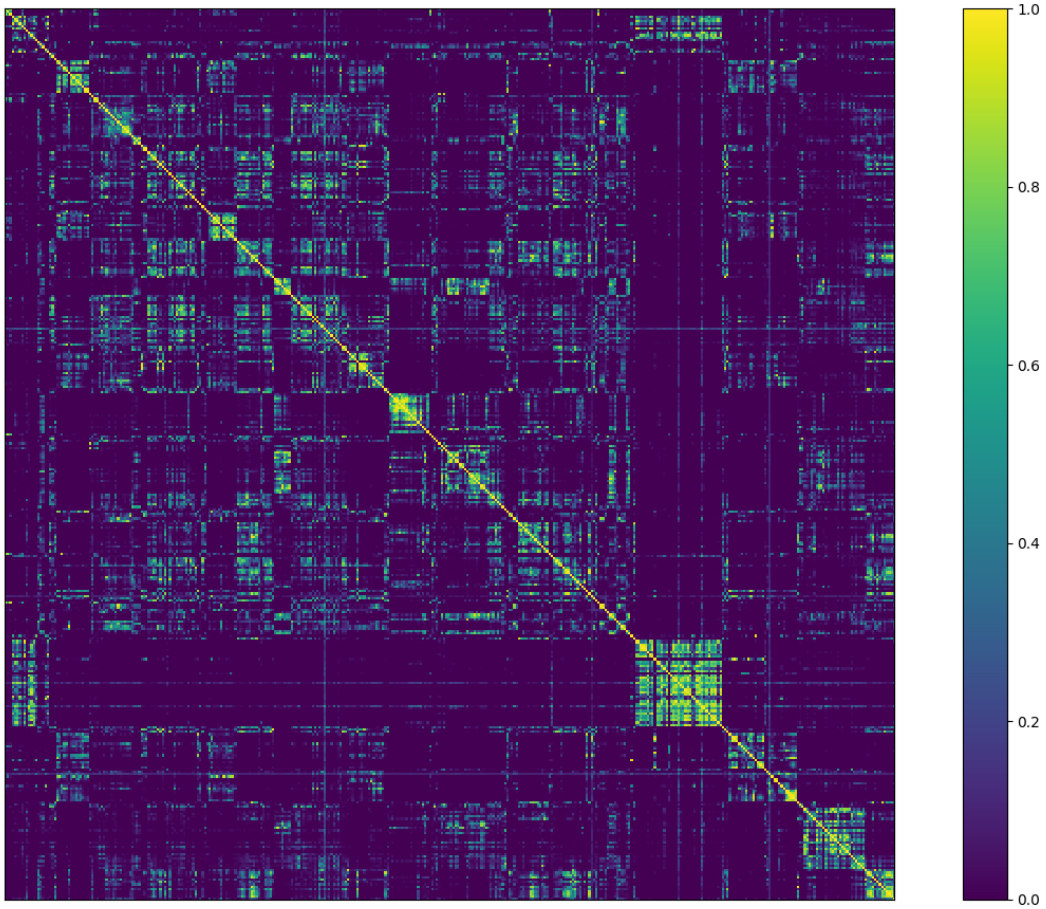



FIGURE 5.16 – Représentation graphique des scores des paires de fragments.

La Figure. 5.17 représente la même matrice, avec deux agrandissements sur des régions qui correspondent aux fragments issus des cartonnages **1328** et **1317**. On peut voir une grande quantité de scores élevés sur l'agrandissement *b*, ce qui indique qu'une grande partie des fragments issus ce cartonnage proviennent en fait peut-être aussi du même document d'après les prédictions de notre modèle. Sur l'agrandissement *a*, on voit que la partie supérieure gauche contient aussi beaucoup de scores élevés, mais beaucoup moins sur le reste de la figure. Cela pourrait indiquer qu'une partie des fragments de ce cartonnage proviennent du même document, mais que les autres proviennent d'une variété d'autres documents, puisqu'ils ne sont même pas corrélés entre eux.

Un autre cas intéressant est celui identifié sur la Figure. 5.17 par la lettre *c*. Ce "bloc" se trouve contenir peu de fragments dont la partie numérique du numéro d'inventaire est la même, mais beaucoup de fragments de numéros d'inventaires contigus (ex. 1567, 1568, 1569 ...). Cela est peut-être la conséquence d'une de ces exceptions dont nous faisons mention au début de cette section, ce qui voudrait dire que beaucoup de ces fragments appartiennent en fait au même cartonnage. Dans tous

les cas, le fait que les numéros d'inventaire de ces fragments soient contigus indique qu'ils ont été découverts proches dans le temps, et donc probablement proches en termes de cartonnages, ce qui augmente les chances qu'ils proviennent des mêmes documents. Notre modèle a donc bien identifié une corrélation entre proximité des numéros d'inventaire et similarité des fragments, en n'ayant bien sûr eu accès qu'aux images elles-mêmes à l'entraînement et à l'inférence.

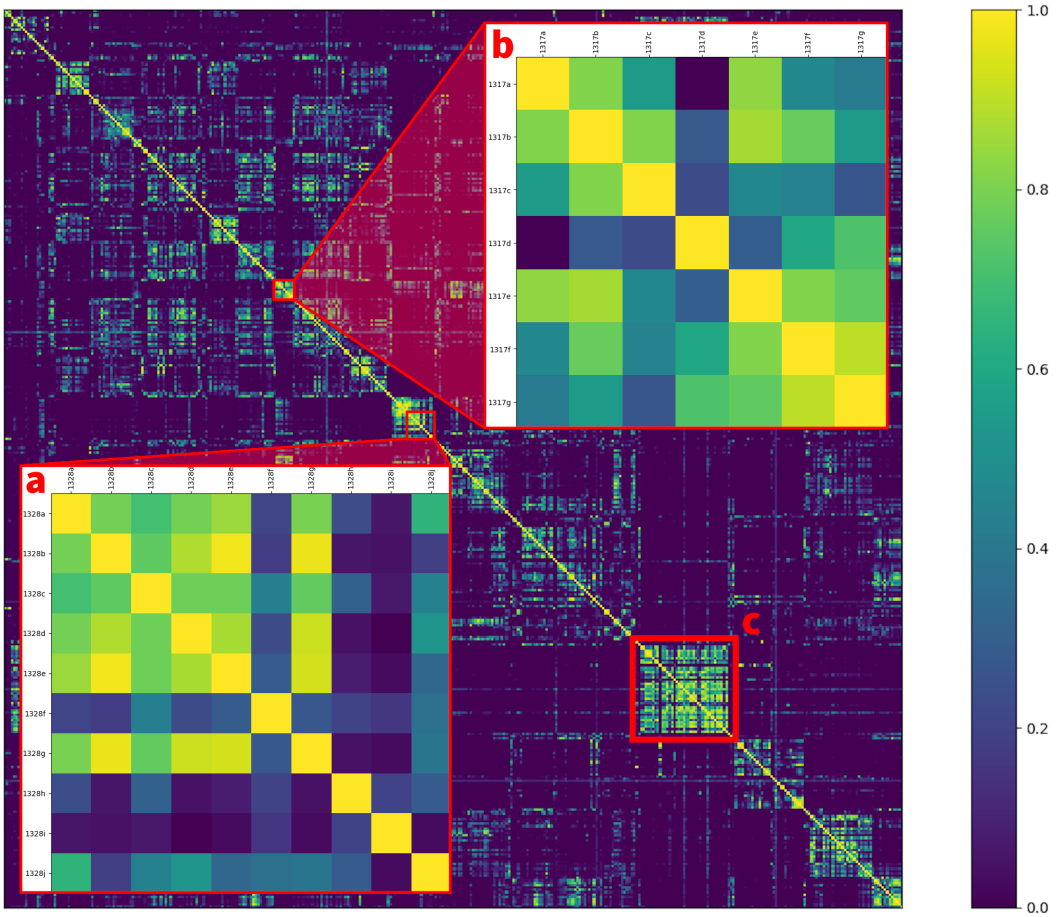



FIGURE 5.17 – Représentation graphique des scores des paires de fragments. Visualisation des fragments issus des cartonnages **1328** (a) et **1317** (b).

# Conclusion

Dans ce chapitre, nous avons dans un premier décrit les caractéristiques des papyrus de la collection *Jouguet* et précisé le protocole d'acquisition numérique qui a été mis en place pour numériser ces fragments. Nous avons ensuite présenté le logiciel *Orio*, qui permet aux papyrologues de manipuler ces fragments numérisés, ainsi que de consulter les propositions d'appairages issues de notre modèle. Puis, nous

avons décrit le protocole d'apprentissage permettant d'obtenir un modèle capable de calculer ces propositions d'appairages sur la collection *Jouguet* sans avoir recours à aucune annotation. Nous avons proposé en collaboration avec les papyrologues du projet *GESHAEM* une étude des suggestions d'appairages générées à travers différents cas de figure, allant des mauvais appairages aux succès. Enfin, nous avons proposé une analyse de ces résultats à travers la notion de cartonnages et d'une hypothèse de travail utilisée par les papyrologues pour identifier les fragments provenant probablement de mêmes documents.

# Chapitre 6

# Discussion et perspectives

On peut imaginer différents axes d'amélioration à toutes les étapes de la méthode que nous avons proposée dans ce manuscrit. Nous décrivons ici les perspectives de recherche futures en fonction des différentes thématiques notre approche.

## Méthode pour extraire une similarité entre fragments à partir de similarités entre patchs

Nous avons expliqué dans la Section. 3.2.4 que nous calculons la moyenne des scores de similarité des paires de patchs entre deux fragments pour calculer la similarité entre deux fragments. Avec un nombre important de patchs extraits par fragments, il pourrait être intéressant d'utiliser des méthodes plus sophistiquées avec par exemple du filtrage des valeurs aberrantes comme *RANSAC* (Fischler et Bolles (1981)). On pourrait aussi imaginer un système de vote majoritaire, ou d'autres mécanismes de consensus. On pourrait aussi calculer une notion de "confiance" en ce score final en regardant l'écart type des scores à partir desquels il a été calculé. Une autre façon de consolider ces scores de similarité entre fragments serait d'étudier la *cohérence* des scores sur un ensemble de fragments. Par exemple, si un fragment $A$ est associé à un fragment $B$, et que ce dernier est associé à un fragment $C$, le fragment $A$ devrait lui aussi être associé au fragment $C$. Si c'est le cas, la confiance que l'on peut attribuer à ces associations est renforcée, alors que si ce n'est pas le cas, elle est réduite.

## Extraction de patchs lors de l'inférence

Concernant les patchs, nous avons expliqué dans la Section. 3.2.2 qu'il était nécessaire lors de l'entraînement en mode *auto-supervisé* d'extraire des patchs qui ne se chevauchent pas pour le bon apprentissage du modèle. Dans tout le manuscrit, nous avons conservé cette règle lors de l'inférence. Elle ne pose pas vraiment problème, mais elle n'a plus lieu d'être lors de l'inférence, car les patchs ne peuvent plus provenir du même fragment. On pourrait donc lever cette restriction et extraire un plus grand nombre de patchs. Cela a potentiellement le pouvoir d'améliorer la qualité des résultats, au prix d'un plus grand temps de calcul.

## Utilisation d'autres architectures de réseaux de neurones

Il serait bien entendu intéressant d'étudier l'apport que peuvent constituer les réseaux triplets, décrits dans la Section. 2.4.3, sur les performances de notre approche, ainsi que l'apport des différentes fonctions de coût qui existent pour optimiser des réseaux siamois ou triplets. Ce serait aussi une piste intéressante pour une exploration plus en profondeur des procédures de sélection d'exemples difficiles dans les batchs combinées à de l'augmentation de données que nous avons abordée dans la Section. 4.4.

## Aller plus loin dans la sélection et l'augmentation d'exemples difficiles dans les batchs

Nous avons vu que sélectionner les exemples difficiles dans les batchs d'entraînement pouvait être une stratégie efficace pour améliorer la convergence des réseaux de neurones. Il serait intéressant de pousser cette idée plus loin en générant des exemples difficiles synthétiques dynamiquement lors de l'entraînement. En s'inspirant de *AutoAugment* (Cubuk et al. (2019)) on pourrait imaginer une fonction paramétrée qui génère des exemples difficiles à partir de ceux déjà existants dans un batch en maximisant leurs difficultés à chaque étape de l'entraînement. Cette stratégie pourrait exploiter les GAN pour générer des données qui optimisent la difficulté des exemples dynamiquement.

## Interprétabilité des résultats obtenus

Notre modèle est capable de produire des propositions d'appairage, mais on ne sait pour l'instant pas expliquer *pourquoi* il a choisi ces propositions. Être capables de donner des éléments d'explication des raisons qui ont fait que le modèle a effectué une prédiction serait une information précieuse pour discuter avec les papyrologues des résultats de la méthode. C'est cependant quelque chose de difficile, car les réseaux de neurones profonds sont encore aujourd'hui en grande partie des *boites noires* et un champ de recherche entier travaille à les rendre plus explicables. Une approche qui a émergé ces dernières années est d'extraire des cartes de chaleur des éléments de l'image d'entrée qui ont le plus contribué à la prédiction (ce que fait *Grad-CAM* par exemple (Selvaraju et al. (2017))). Cette approche n'est pas directement applicable aux réseaux siamois, mais d'autres approches commencent à apparaître dans la littérature pour le cas précis des réseaux siamois (Utkin et al. (2021)).

## Prise en compte des connaissances métiers : aller vers de l'apprentissage multimodal

Lors de nos expériences sur la collection *Jouguet*, nous avons considéré les rectos et les versos des fragments de manière indépendante. C'est-à-dire que pour un frag-

ment donné, on produit en réalité deux listes classées de suggestions de fragments, une pour le recto et une pour le verso. Ces deux listes peuvent être combinées pour obtenir un score global pour le fragment, mais il serait peut-être plus efficace de directement concevoir une architecture de réseaux siamois qui intègre le recto et le verso de chaque fragment.

De même, nous n'avons exploité ni les images infrarouges, ni les méta-données qui étaient disponibles dans nos expériences. Ces dernières peuvent servir à faire un tri *a posteriori* en filtrant les appairages impossibles, mais elles pourraient être encore plus utiles en les intégrant directement aux données d'entrées de nos modèles siamois pour en faire une approche multimodale. Une architecture comme celle proposée par Audebert et al. (2019) peut par exemple permettre de projeter des données hétérogènes dans un espace latent commun, ce qui nous permettrait d'exploiter les méta données fournies par les papyrologues.

## Reconstruction automatique des documents

Enfin, nous avons travaillé durant cette thèse sur la suggestion d'appairages de fragments, mais pas sur la reconstruction "physique" des documents. Idéalement, on voudrait pouvoir directement reconstruire les documents automatiquement. Une piste intéressante est celle proposée par Ostertag et Beurton-Aimar (2021) qui utilise un *Graph Neural Network* pour prédire les positions relatives de patchs de fragments de papyrus.

Cet objectif est particulièrement difficile avec des papyrus. En effet, à cause de la grande dégradation des fragments, des fibres qui se chevauchent lorsqu'ils sont déchirés, l'exploitation du contour est quasiment impossible. De même, des fragments constituant un document peuvent simplement avoir disparu. Une exploitation de la structure du texte (et peut-être même de son contenu) et des caractéristiques fines des fibres des papyrus serait utile pour atteindre cet objectif.

# Conclusion

Durant ces travaux de thèse, nous avons développé une approche d'apprentissage profond basée sur des réseaux convolutifs siamois afin de calculer des suggestions d'appairages de fragments de documents anciens. Nous avons proposé une approche d'entraînement *auto-supervisée* permettant de s'abstraire de la nécessité d'avoir à disposition des données annotées. En combinant cette approche avec de l'apprentissage par transfert en pré-entraînant le modèle sur un autre jeu de données de documents, nous avons pu atteindre par exemple un score de 75% de *top-1 accuracy* sur la base *Michigan*. Cela signifie qu'en moyenne, il y a trois chances sur quatre que le fragment classé en première position par rapport au fragment requête soit effectivement un fragment qui est originaire du même document. Nos méthodes ont été entraînées et évaluées sur deux jeux de données internationaux et publiques. Le jeu de données de fragments de documents anciens synthétique *Hisfrag* et un jeu de données de papyrus que nous avons conçu à partir de la collection de papyrus de l'Université du Michigan en sélectionnant et pré-traitant les images pour les rendre directement utilisables dans un contexte d'apprentissage automatique.

En collaboration avec les papyrologues du projet *GESHAEM*, nous avons étudié la pertinence des suggestions d'appairage que propose un modèle entraîné avec notre approche *auto-supervisée* sur les papyrus de la collection *Jouguet*. Malgré la difficulté de ce jeu de données, les propositions d'assemblage calculées par notre modèle se sont souvent révélées pertinentes et nous avons pu identifier que nos prédictions sont cohérentes avec une hypothèse de travail utilisée par les papyrologues concernant un lien entre le raccord de fragments et leurs appartenance au même cartonnage. Un logiciel appelé *Orio* a été développé au sein du projet *GESHAEM* pour permettre aux papyrologues de manipuler des fragments de papyrus numériques et d'accéder plus facilement aux propositions d'assemblages calculées par notre modèle.

Ces travaux ont donné lieux à deux publications, l'une dans un workshop de la conférence *ICDAR* (*HIP*) et l'autre dans le journal *IJDAR*. Cette dernière publication nous a aussi donné l'opportunité de présenter nos travaux dans la *main conference* de *ICDAR* en 2021. Les avancements de nos travaux ont aussi été présentés à différentes occasions dans des séminaires nationaux (*SIFED*) et lors de journées de travail avec des chercheurs en humanités numériques à l'international (*Bâle* et *Heidelberg*).

Le projet *GESHAEM* va se poursuivre pendant encore deux ans après la fin de cette thèse. Nous espérons que les méthodes qui ont été développées au cours

ces trois ans de recherche seront utiles aux papyrologues et accéléreront ainsi le processus de reconstruction des fragments de papyrus de la collection *Jouguet*.

# Bibliographie


[Abitbol et al. 2021] Abitbol, Roy ; Shimshoni, Ilan ; Ben-Dov, Jonathan : Machine Learning Based Assembly of Fragments of Ancient Papyrus. Dans : *Journal on Computing and Cultural Heritage (JOCCH)* 14 (2021), n. 3, pp. 1–21 16, 54

[Alkhawlani et al. 2015] Alkhawlani, Mohammed ; Elmogy, Mohammed ; El Bakry, Hazem : Text-based, content-based, and semantic-based image retrievals: A survey. Dans : *Int. J. Comput. Inf. Technol* 4 (2015), n. 01, pp. 58–66 54

[Anava et al. 2020] Anava, Sarit ; Neuhof, Moran ; Gingold, Hila ; Sagy, Or ; Munters, Arielle ; Svensson, Emma M. ; Afshinnekoo, Ebrahim ; Danko, David ; Foox, Jonathan ; Shor, Pnina et al. : Illuminating genetic mysteries of the dead sea scrolls. Dans : *Cell* 181 (2020), n. 6, pp. 1218–1231 17

[Atanasiu et Marthot-Santaniello 2021] Atanasiu, Vlad ; Marthot-Santaniello, Isabelle : Legibility Enhancement of Papyri Using Color Processing and Visual Illusions: A Case Study in Critical Vision. Dans : *arXiv preprint arXiv:2104.01106* (2021) 15

[Audebert et al. 2019] Audebert, Nicolas ; Herold, Catherine ; Slimani, Kuider ; Vidal, Cédric : Multimodal deep networks for text and image-based document classification. Dans : *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* Springer (Veranst.), 2019, pp. 427–443 122

[Bagnall 2011] Bagnall, Roger S. : *The Oxford handbook of papyrology*. Oxford University Press, 2011 6, 7

[Bahdanau et al. 2014] Bahdanau, Dzmitry ; Cho, Kyunghyun ; Bengio, Yoshua : Neural machine translation by jointly learning to align and translate. Dans : *arXiv preprint arXiv:1409.0473* (2014) 68

[Bellet et al. 2015] Bellet, Aurélien ; Habrard, Amaury ; Sebban, Marc : Metric learning. Dans : *Synthesis Lectures on Artificial Intelligence and Machine Learning* 9 (2015), n. 1, pp. 1–151 38

[Bhattacharyya 1943] Bhattacharyya, Anil : On a measure of divergence between two statistical populations defined by their probability distributions. Dans : *Bull. Calcutta Math. Soc.* 35 (1943), pp. 99–109 39

[Biswas et al. 2005] Biswas, Arindam ; Bhowmick, Partha ; Bhattacharya, Bhargab B. : Reconstruction of torn documents using contour maps. Dans : *IEEE International Conference on Image Processing 2005* 3 IEEE (Veranst.), 2005, pp. III–517 52

[Bondi et al. 2017] Bondi, Luca ; Güera, David ; Baroffio, Luca ; Bestagini, Paolo ; Delp, Edward J. ; Tubaro, Stefano : A preliminary study on convolutional neural networks for camera model identification. Dans : *Electronic Imaging* 2017 (2017), n. 7, pp. 67–76 56

[Boser et al. 1992] Boser, Bernhard E. ; Guyon, Isabelle M. ; Vapnik, Vladimir N. : A training algorithm for optimal margin classifiers. Dans : *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152 23

[Bromley et al. 1993] Bromley, Jane ; Bentz, James W. ; Bottou, Léon ; Guyon, Isabelle ; LeCun, Yann ; Moore, Cliff ; Säckinger, Eduard ; Shah, Roopak : Signature verification using a "siamese" time delay neural network. Dans : *International Journal of Pattern Recognition and Artificial Intelligence* 7 (1993), n. 04, pp. 669–688 42

[Capobianco et Marinai 2017] Capobianco, Samuele ; Marinai, Simone : Docemul: a toolkit to generate structured historical documents. Dans : *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)* 1 IEEE (Veranst.), 2017, pp. 1186–1191 95

[Chang et al. 2017] Chang, Haw-Shiuan ; Learned-Miller, Erik ; McCallum, Andrew : Active bias: Training more accurate neural networks by emphasizing high variance samples. Dans : *Advances in Neural Information Processing Systems* 30 (2017), pp. 1002–1012 98

[Chen et al. 2020] Chen, Chen ; Qin, Chen ; Qiu, Huaqi ; Ouyang, Cheng ; Wang, Shuo ; Chen, Liang ; Tarroni, Giacomo ; Bai, Wenjia ; Rueckert, Daniel : Realistic adversarial data augmentation for MR image segmentation. Dans : *International Conference on Medical Image Computing and Computer-Assisted Intervention* Springer (Veranst.), 2020, pp. 667–677 94

[Chen et al. 2017] Chen, Weihua ; Chen, Xiaotang ; Zhang, Jianguo ; Huang, Kaiqi : Beyond triplet loss: a deep quadruplet network for person re-identification. Dans : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 403–412 43

[Christlein et al. 2019] Christlein, Vincent ; Nicolaou, Anguelos ; Seuret, Mathias ; Stutzmann, Dominique ; Maier, Andreas : *ICDAR 2019 Competition on Image Retrieval for Historical Handwritten Documents*. 2019 66

[Cilia et al. 2021] Cilia, Nicole D. ; De Stefano, Claudio ; Fontanella, Francesco ; Marthot-Santaniello, Isabelle ; Freca, Alessandra Scotto di : PapyRow: A Dataset of Row Images from Ancient Greek Papyri for Writers Identification. Dans : Del Bimbo, Alberto (Hrsg.) ; Cucchiara, Rita (Hrsg.) ; Sclaroff, Stan (Hrsg.) ; Farinella, Giovanni M. (Hrsg.) ; Mei, Tao (Hrsg.) ; Bertini, Marco (Hrsg.) ; Escalante, Hugo J. (Hrsg.) ; Vezzani, Roberto (Hrsg.): *Pattern Recognition. ICPR International Workshops and Challenges*. Cham : Springer International Publishing, 2021, pp. 223–234. – ISBN 978-3-030-68787-8 8, 19

[Cocosco et al. 1997] Cocosco, Chris A. ; Kollokian, Vasken ; Kwan, Remi K-S ; Pike, G B. ; Evans, Alan C. : Brainweb: Online interface to a 3D MRI simulated brain database. Dans : *NeuroImage* Citeseer (Veranst.), 1997 94

[Cubuk et al. 2019] Cubuk, Ekin D. ; Zoph, Barret ; Mane, Dandelion ; Vasudevan, Vijay ; Le, Quoc V. : Autoaugment: Learning augmentation strategies from data. Dans : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 113–123 121

[Célor 2018] Célor, Pierre : *Aide à la reconstruction de fragments de papyrus*, Université de Bordeaux, Mémoire de Master, 2018 107

[De Smet 2008] De Smet, Patrick : Reconstruction of ripped-up documents using fragment stack analysis procedures. Dans : *Forensic science international* 176 (2008), n. 2-3, pp. 124–136 52

[Deever et Gallagher 2012] Deever, Aaron ; Gallagher, Andrew : Semi-automatic assembly of real cross-cut shredded documents. Dans : *2012 19th IEEE International Conference on Image Processing* IEEE (Veranst.), 2012, pp. 233–236 52

[Deng et al. 2019] Deng, Jiankang ; Guo, Jia ; Xue, Niannan ; Zafeiriou, Stefanos : Arcface: Additive angular margin loss for deep face recognition. Dans : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699 43

[Deshpande et al. 2015] Deshpande, Aditya ; Rock, Jason ; Forsyth, David : Learning large-scale automatic image colorization. Dans : *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 567–575 34

[Diem et al. 2010] Diem, Markus ; Kleber, Florian ; Sablatnig, Robert : Document analysis applied to fragments: feature set for the reconstruction of torn documents. Dans : *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, 2010, pp. 393–400 52

[Fischler et Bolles 1981] Fischler, Martin A. ; Bolles, Robert C. : Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Dans : *Communications of the ACM* 24 (1981), n. 6, pp. 381–395 120

[Fix et Hodges 1951] Fix, Evelyn ; Hodges, Joseph L. : Discriminatory analysis. Nonparametric discrimination: Consistency properties. Dans : *International Statistical Review/Revue Internationale de Statistique* 57 (1951), n. 3, pp. 238–247 23

[da Gama Leitao et Stolfi 2002] Gama Leitao, Helena C. da ; Stolfi, Jorge : A multiscale method for the reassembly of two-dimensional fragmented objects. Dans : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), n. 9, pp. 1239–1251 53

[Gao et Jojic 2016] Gao, Tianxiang ; Jojic, Vladimir : Sample importance in training deep neural networks. (2016) 97, 98

[Gascou 2001] Gascou, Jean : La papyrologie et l'imagerie numérique. Dans : *Sauvegarde et diffusion du patrimoine scientifique et culturel à Strasbourg: l'apport des technologies de l'information et de la communication.* Carré Blanc éditions, 2001, pp. p. 71–77. – URL https://halshs.archives-ouvertes.fr/halshs-00001532 5

[Goodfellow et al. 2016] Goodfellow, Ian ; Bengio, Yoshua ; Courville, Aaron : *Deep learning.* MIT press, 2016 27, 35, 36, 37, 94

[Grüning et al. 2019] Grüning, Tobias ; Leifert, Gundram ; Strauss, Tobias ; Michael, Johannes ; Labahn, Roger : A two-stage method for text line detection in historical documents. Dans : *International Journal on Document Analysis and Recognition (IJDAR)* 22 (2019), n. 3, pp. 285–302 16, 67, 69

[Hadsell et al. 2006] Hadsell, Raia ; Chopra, Sumit ; LeCun, Yann : Dimensionality reduction by learning an invariant mapping. Dans : *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)* 2 IEEE (Veranst.), 2006, pp. 1735–1742 42

[Haliassos et al. 2020] Haliassos, Alexandros ; Barmpoutis, Panagiotis ; Stathaki, Tania ; Quirke, Stephen ; Constantinides, Anthony : *Classification and Detection of Symbols in Ancient Papyri.* 2020 17

[He et al. 2015] He, Kaiming ; Zhang, Xiangyu ; Ren, Shaoqing ; Sun, Jian : Deep Residual Learning for Image Recognition. Dans : *CoRR* abs/1512.03385 (2015). – URL http://arxiv.org/abs/1512.03385 31

[He et al. 2016] He, Sheng ; Samara, Petros ; Burgers, Jan ; Schomaker, Lambert : Image-based historical manuscript dating using contour and stroke fragments. Dans : *Pattern Recognition* 58 (2016), pp. 159–171 18

[He et Schomaker 2020] He, Sheng ; Schomaker, Lambert : Fragnet: Writer identification using deep fragment networks. Dans : *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 3013–3022 18

[Horiguchi et al. 2016] Horiguchi, Shota ; Ikami, Daiki ; Aizawa, Kiyoharu : Significance of softmax-based features over metric learning-based features. (2016) 41

[Hornik et al. 1989] Hornik, Kurt ; Stinchcombe, Maxwell ; White, Halbert : Multilayer feedforward networks are universal approximators. Dans : *Neural networks* 2 (1989), n. 5, pp. 359–366 26

[Iizuka et al. 2016] Iizuka, Satoshi ; Simo-Serra, Edgar ; Ishikawa, Hiroshi : Let there be color! Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. Dans : *ACM Transactions on Graphics (ToG)* 35 (2016), n. 4, pp. 1–11 34

[Ioffe et Szegedy 2015] Ioffe, Sergey ; Szegedy, Christian : Batch normalization: Accelerating deep network training by reducing internal covariate shift. Dans : *International conference on machine learning* PMLR (Veranst.), 2015, pp. 448–456 32

[Jing et Tian 2020] Jing, Longlong ; Tian, Yingli : Self-supervised visual feature learning with deep neural networks: A survey. Dans : *IEEE transactions on pattern analysis and machine intelligence* (2020) 34

[Johnson 1967] Johnson, Stephen C. : Hierarchical clustering schemes. Dans : *Psychometrika* 32 (1967), n. 3, pp. 241–254 24

[Journet et al. 2017] Journet, Nicholas ; Visani, Muriel ; Mansencal, Boris ; Van-Cuong, Kieu ; Billy, Antoine : Doccreator: A new software for creating synthetic ground-truthed document images. Dans : *Journal of imaging* 3 (2017), n. 4, pp. 62 94, 95

[KAYA 2019] KAYA, H.Ş : Deep Metric Learning: A Survey. Dans : *Symmetry* (2019) 40, 41, 44

[Kha Vu 2021] Kha Vu, Chan : *Deep Metric Learning: A (Long) Survey*. 2021. – URL https://hav4ik.github.io/articles/deep-metric-learning-survey 43

[Kingma et Ba 2014] Kingma, Diederik P. ; Ba, Jimmy : Adam: A method for stochastic optimization. Dans : *arXiv preprint arXiv:1412.6980* (2014) 73

[Kišš et al. 2021] Kišš, Martin ; Beneš, Karel ; Hradiš, Michal : AT-ST: Self-Training Adaptation Strategy for OCR in Domains with Limited Transcriptions. Dans : *arXiv preprint arXiv:2104.13037* (2021) 17

[Koch 2015] Koch, Gregory R. : Siamese Neural Networks for One-Shot Image Recognition, 2015 70, 71

[Krizhevsky et al. 2012] Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E. : Imagenet classification with deep convolutional neural networks. Dans : *Advances in neural information processing systems* 25 (2012), pp. 1097–1105 30

[Kullback et Leibler 1951] Kullback, Solomon ; Leibler, Richard A. : On information and sufficiency. Dans : *The annals of mathematical statistics* 22 (1951), n. 1, pp. 79–86 39

[Lake et al. 2015] Lake, Brenden M. ; Salakhutdinov, Ruslan ; Tenenbaum, Joshua B. : Human-level concept learning through probabilistic program induction. Dans : *Science* 350 (2015), n. 6266, pp. 1332–1338. – ISSN 0036-8075 71

[Larsson et al. 2016] Larsson, Gustav ; Maire, Michael ; Shakhnarovich, Gregory : Learning representations for automatic colorization. Dans : *European conference on computer vision* Springer (Veranst.), 2016, pp. 577–593 34

[Le et Li 2019] Le, Canyu ; Li, Xin : JigsawNet: Shredded image reassembly using convolutional neural network and loop-based composition. Dans : *IEEE Transactions on Image Processing* 28 (2019), n. 8, pp. 4000–4015 51

[LeCun et al. 1989] LeCun, Yann ; Boser, Bernhard ; Denker, John S. ; Henderson, Donnie ; Howard, Richard E. ; Hubbard, Wayne ; Jackel, Lawrence D. : Backpropagation applied to handwritten zip code recognition. Dans : *Neural computation* 1 (1989), n. 4, pp. 541–551 28

[LeCun et al. 1998] LeCun, Yann ; Bottou, Léon ; Bengio, Yoshua ; Haffner, Patrick : Gradient-based learning applied to document recognition. Dans : *Proceedings of the IEEE* 86 (1998), n. 11, pp. 2278–2324 28

[Lehenmeier et al. 2020] Lehenmeier, Constantin ; Burghardt, Manuel ; Mischka, Bernadette : Layout Detection and Table Recognition–Recent Challenges in Digitizing Historical Documents and Handwritten Tabular Data. Dans : *International Conference on Theory and Practice of Digital Libraries* Springer (Veranst.), 2020, pp. 229–242 16

[Liu et Deng 2015] Liu, S. ; Deng, W. : Very deep convolutional neural network based image classification using small training sample size. Dans : *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015, pp. 730–734 31, 71

[Liu et al. 2017] Liu, Weiyang ; Wen, Yandong ; Yu, Zhiding ; Li, Ming ; Raj, Bhiksha ; Song, Le : Sphereface: Deep hypersphere embedding for face recognition. Dans : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212–220 43

[Lombardi et Marinai 2020] Lombardi, Francesco ; Marinai, Simone : Deep Learning for Historical Document Analysis and Recognition—A Survey. Dans : *Journal of Imaging* 6 (2020), n. 10, pp. 110 15

[MacQueen et al. 1967] MacQueen, James et al. : Some methods for classification and analysis of multivariate observations. Dans : *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* 1 Oakland, CA, USA (Veranst.), 1967, pp. 281–297 24

[Mahalanobis 1936] Mahalanobis, Prasanta C. : On the generalized distance in statistics National Institute of Science of India (Veranst.), 1936 39

[Marques et Freitas 2009] Marques, Marlos A. ; Freitas, Cinthia O. : Reconstructing strip-shredded documents using color as feature matching. Dans : *Proceedings of the 2009 ACM symposium on Applied Computing*, 2009, pp. 893–894 52

[Masi et al. 2018] Masi, Iacopo ; Wu, Yue ; Hassner, Tal ; Natarajan, Prem : Deep face recognition: A survey. Dans : *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)* IEEE (Veranst.), 2018, pp. 471–478 39

[McCulloch et Pitts 1943] McCulloch, Warren S. ; Pitts, Walter : A logical calculus of the ideas immanent in nervous activity. Dans : *The bulletin of mathematical biophysics* 5 (1943), n. 4, pp. 115–133 26

[Nasir et Siddiqi 2021] Nasir, Sidra ; Siddiqi, Imran : Learning Features for Writer Identification from Handwriting on Papyri. Dans : Djeddi, Chawki (Hrsg.) ; Kessentini, Yousri (Hrsg.) ; Siddiqi, Imran (Hrsg.) ; Jmaiel, Mohamed (Hrsg.): *Pattern Recognition and Artificial Intelligence*. Cham : Springer International Publishing, 2021, pp. 229–241. – ISBN 978-3-030-71804-6 8

[Nasir et al. 2021] Nasir, Sidra ; Siddiqi, Imran ; Moetesum, Momina : Writer Characterization from Handwriting on Papyri Using Multi-step Feature Learning. Dans : *International Conference on Document Analysis and Recognition* Springer (Veranst.), 2021, pp. 451–465 18

[Nguyen et al. 2021a] Nguyen, Ngoc L. ; Anger, Jeremy ; Davy, Axel ; Arias, Pablo ; Facciolo, Gabriele : Self-Supervised Multi-Image Super-Resolution for Push-Frame Satellite Images. Dans : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1121–1131 34

[Nguyen et al. 2021b] Nguyen, Tien-Nam ; Burie, Jean-Christophe ; Le, Thi-Lan ; Schweyer, Anne-Valerie : On the Use of Attention in Deep Learning Based Denoising Method for Ancient Cham Inscription Images. Dans : *International Conference on Document Analysis and Recognition* Springer (Veranst.), 2021, pp. 400–415 15

[Nielsen 2015] Nielsen, Michael A. : *Neural Networks and Deep Learning*. Determination Press, 2015 37

[Noroozi et Favaro 2016] Noroozi, Mehdi ; Favaro, Paolo : Unsupervised learning of visual representations by solving jigsaw puzzles. Dans : *European conference on computer vision* Springer (Veranst.), 2016, pp. 69–84 34

[Oates et al. 1999] Oates, John F. ; Weinberg, Richard J. ; Sosin, Joshua D. ; Johnson, Paul B. : Reading invisible ink: digital imaging of P. Duk. Inv. 716. Dans : *Zeitschrift für Papyrologie und Epigraphik* (1999), pp. 127–130 8

[Ostertag et Beurton-Aimar 2020] Ostertag, Cecilia ; Beurton-Aimar, Marie : Matching ostraca fragments using a siamese neural network. Dans : *Pattern Recognition Letters* 131 (2020), pp. 336–340 16

[Ostertag et Beurton-Aimar 2021] Ostertag, Cecilia ; Beurton-Aimar, Marie : Using Graph Neural Networks to Reconstruct Ancient Documents. Dans : *International Conference on Pattern Recognition* Springer (Veranst.), 2021, pp. 39–53 16, 122

[Paixao et al. 2020] Paixao, Thiago M. ; Berriel, Rodrigo F. ; Boeres, Maria ; Koerich, Alessandro L. ; Badue, Claudine ; Souza, Alberto F D. ; Oliveira-Santos, Thiago : Fast (er) reconstruction of shredded text documents via self-supervised deep asymmetric metric learning. Dans : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14343–14351 52

[Paixão et al. 2020] Paixão, Thiago M. ; Berriel, Rodrigo F. ; Boeres, Maria C. ; Koerich, Alessandro L. ; Badue, Claudine ; De Souza, Alberto F. ; Oliveira-Santos, Thiago : Self-supervised deep reconstruction of mixed strip-shredded text documents. Dans : *Pattern Recognition* 107 (2020), pp. 107535 52, 53

[Papaodysseus et al. 2002] Papaodysseus, Constantin ; Panagopoulos, Thanasis ; Exarhos, Michael ; Triantafillou, Constantin ; Fragoulis, Dimitrios ; Doumas, Christos : Contour-shape based reconstruction of fragmented, 1600 bc wall paintings. Dans : *IEEE Transactions on Signal Processing* 50 (2002), n. 6, pp. 1277–1288 53

[Paumard et al. 2018] Paumard, Marie-Morgane ; Picard, David ; Tabia, Hedi : Jigsaw puzzle solving using local feature co-occurrences in deep neural networks. Dans : *2018 25th IEEE International Conference on Image Processing (ICIP)* IEEE (Veranst.), 2018, pp. 1018–1022 51

[Philips et Tabrizi 2020] Philips, James P. ; Tabrizi, Nasseh : Historical Document Processing: Historical Document Processing: A Survey of Techniques, Tools, and Trends. Dans : *arXiv preprint arXiv:2002.06300* (2020) 14

[Pirrone et al. 2019] Pirrone, Antoine ; Beurton-Aimar, Marie ; Journet, Nicholas : Papy-S-Net: A Siamese Network to match papyrus fragments. Dans : *Proceedings of the 5th International Workshop on Historical Document Imaging and Processing*, 2019, pp. 78–83 16, 50, 70

[Pirrone et al. 2021] Pirrone, Antoine ; Beurton-Aimar, Marie ; Journet, Nicholas : Self-supervised deep metric learning for ancient papyrus fragments retrieval. Dans : *International Journal on Document Analysis and Recognition (IJDAR)* (2021), pp. 1–16 16, 50, 82

[Pondenkandath et al. 2019] Pondenkandath, Vinaychandran ; Alberti, Michele ; Diatta, Michaël ; Ingold, Rolf ; Liwicki, Marcus : Historical document synthesis with generative adversarial networks. Dans : *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)* 5 IEEE (Veranst.), 2019, pp. 146–151 95

[Popović et al. 2021] Popović, Mladen ; Dhali, Maruf A. ; Schomaker, Lambert : Artificial intelligence based writer identification generates new evidence for the unknown scribes of the Dead Sea Scrolls exemplified by the Great Isaiah Scroll (1QIsaa). Dans : *PloS one* 16 (2021), n. 4, pp. e0249769 8

[Reggiani 2017] Reggiani, Nicola : *Digital papyrology.* De Gruyter, 2017 12

[Richter et al. 2014] Richter, Fabian ; Ries, Christian X. ; Lienhart, Rainer : Evaluation of discriminative models for the reconstruction of hand-torn documents. Dans : *Asian Conference on Computer Vision* Springer (Veranst.), 2014, pp. 671–686 52

[Romero-Ramirez et al. 2018] Romero-Ramirez, Francisco J. ; Muñoz-Salinas, Rafael ; Medina-Carnicer, Rafael : Speeded up detection of squared fiducial markers. Dans : *Image and vision Computing* 76 (2018), pp. 38–47 106

[Ronneberger et al. 2015] Ronneberger, Olaf ; Fischer, Philipp ; Brox, Thomas : U-net: Convolutional networks for biomedical image segmentation. Dans : *International Conference on Medical image computing and computer-assisted intervention* Springer (Veranst.), 2015, pp. 234–241 68

[Rosten et Drummond 2006] Rosten, Edward ; Drummond, Tom : Machine learning for high-speed corner detection. Dans : *European conference on computer vision* Springer (Veranst.), 2006, pp. 430–443 18

[Rumelhart et al. 1986] Rumelhart, David E. ; Hinton, Geoffrey E. ; Williams, Ronald J. : Learning representations by back-propagating errors. Dans : *nature* 323 (1986), n. 6088, pp. 533–536 27

[Saritha et al. 2019] Saritha, R R. ; Paul, Varghese ; Kumar, P G. : Content based image retrieval using deep learning process. Dans : *Cluster Computing* 22 (2019), n. 2, pp. 4187–4200 54

[Schroff et al. 2015] Schroff, Florian ; Kalenichenko, Dmitry ; Philbin, James : FaceNet: A Unified Embedding for Face Recognition and Clustering. Dans : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015 42, 43, 44, 98

[Scius-Bertrand et al. 2021] Scius-Bertrand, Anna ; Jungo, Michael ; Wolf, Beat ; Fischer, Andreas ; Bui, Marc : Annotation-Free Character Detection in Historical Vietnamese Stele Images. Dans : *International Conference on Document Analysis and Recognition* Springer (Veranst.), 2021, pp. 432–447 17

[Selvaraju et al. 2017] Selvaraju, Ramprasaath R. ; Cogswell, Michael ; Das, Abhishek ; Vedantam, Ramakrishna ; Parikh, Devi ; Batra, Dhruv : Grad-cam: Visual explanations from deep networks via gradient-based localization. Dans : *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626 121

[Seuret et al. 2020] Seuret, Mathias ; Nicolaou, Anguelos ; Maier, Andreas ; Christlein, Vincent ; Stutzmann, Dominique : ICFHR 2020 competition on image retrieval for historical handwritten fragments. Dans : *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)* IEEE (Veranst.), 2020, pp. 216–221 65

[Seuret et al. September, 2020] Seuret, Mathias ; Nicolaou, Anguelos ; Stutzmann, Dominique ; Maier, Andreas ; Christlein, Vincent : ICFHR 2020 Competition on Image Retrieval for Historical Handwritten Fragments. Dans : *International Conference on Frontiers of Handwriting Recognition* (September, 2020) 18

[Shrivastava et al. 2016] Shrivastava, Abhinav ; Gupta, Abhinav ; Girshick, Ross : Training region-based object detectors with online hard example mining. Dans : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 761–769 97, 98

[Sparavigna 2009] Sparavigna, Amelia : Digital restoration of ancient papyri. Dans : *arXiv preprint arXiv:0903.5045* (2009) 15

[Srivastava et al. 2014] Srivastava, Nitish ; Hinton, Geoffrey ; Krizhevsky, Alex ; Sutskever, Ilya ; Salakhutdinov, Ruslan : Dropout: a simple way to prevent neural networks from overfitting. Dans : *The journal of machine learning research* 15 (2014), n. 1, pp. 1929–1958 37

[Stauffer et al. 2021] Stauffer, Michael ; Maergner, Paul ; Fischer, Andreas ; Riesen, Kaspar : A survey of state of the art methods employed in the offline signature verification process. Dans : *New Trends in Business Information Systems and Technology*. Springer, 2021, pp. 17–30 40

[Sudarma 2015] Sudarma, Made : Identifying of the Cielab Space Color for the Balinese Papyrus Characters. Dans : *Telkomnika Indonesian Journal of Electrical Engineering* 13 (2015), pp. 321–328 8, 15

[Swindall et al. 2021] Swindall, Matthew I. ; Croisdale, Gregory ; Hunter, Chase C. ; Keener, Ben ; Williams, Alex C. ; Brusuelas, James H. ; Krevans, Nita ; Sellew, Melissa ; Fortson, Lucy ; Wallin, John F. : Exploring Learning Approaches for Ancient Greek Character Recognition with Citizen Science Data. (2021) 8

[Szegedy et al. 2017] Szegedy, Christian ; Ioffe, Sergey ; Vanhoucke, Vincent ; Alemi, Alexander A. : Inception-v4, inception-resnet and the impact of residual connections on learning. Dans : *Thirty-first AAAI conference on artificial intelligence*, 2017 32

[Szegedy et al. 2015] Szegedy, Christian ; Liu, Wei ; Jia, Yangqing ; Sermanet, Pierre ; Reed, Scott ; Anguelov, Dragomir ; Erhan, Dumitru ; Vanhoucke, Vincent ; Rabinovich, Andrew : Going deeper with convolutions. Dans : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9 30

[Szegedy et al. 2016] Szegedy, Christian ; Vanhoucke, Vincent ; Ioffe, Sergey ; Shlens, Jon ; Wojna, Zbigniew : Rethinking the inception architecture for computer vision. Dans : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826 32

[Tan et al. 2018] Tan, Chuanqi ; Sun, Fuchun ; Kong, Tao ; Zhang, Wenchang ; Yang, Chao ; Liu, Chunfang : A survey on deep transfer learning. Dans : *International conference on artificial neural networks* Springer (Veranst.), 2018, pp. 270–279 32

[Utkin et al. 2021] Utkin, Lev ; Kovalev, Maxim ; Kasimov, Ernest : An explanation method for siamese neural networks. Dans : *Proceedings of International Scientific Conference on Telecommunications, Computing and Control* Springer (Veranst.), 2021, pp. 219–230 121

[Wang et al. 2018] Wang, Hao ; Wang, Yitong ; Zhou, Zheng ; Ji, Xing ; Gong, Dihong ; Zhou, Jingchao ; Li, Zhifeng ; Liu, Wei : Cosface: Large margin cosine loss for deep face recognition. Dans : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5265–5274 43

[Wen et al. 2016] Wen, Yandong ; Zhang, Kaipeng ; Li, Zhifeng ; Qiao, Yu : A discriminative feature learning approach for deep face recognition. Dans : *European conference on computer vision* Springer (Veranst.), 2016, pp. 499–515 43

[Wick et Reul 2021] Wick, Christoph ; Reul, Christian : One-Model Ensemble-Learning for Text Recognition of Historical Printings. Dans : *International Conference on Document Analysis and Recognition* Springer (Veranst.), 2021, pp. 385–399 17

[Zhao et al. 2020] Zhao, Can ; Dewey, Blake E. ; Pham, Dzung L. ; Calabresi, Peter A. ; Reich, Daniel S. ; Prince, Jerry L. : SMORE: A self-supervised anti-aliasing and super-resolution algorithm for MRI using deep learning. Dans : *IEEE transactions on medical imaging* 40 (2020), n. 3, pp. 805–817 34

[Zhuang et al. 2020] Zhuang, Fuzhen ; Qi, Zhiyuan ; Duan, Keyu ; Xi, Dongbo ; Zhu, Yongchun ; Zhu, Hengshu ; Xiong, Hui ; He, Qing : A comprehensive survey on transfer learning. Dans : *Proceedings of the IEEE* 109 (2020), n. 1, pp. 43–76 32

# Annexe A

# Liste de bases de données de papyrus et documents anciens numérisés

— Advanced Papyrological Information System (APIS UM)
— Harvard Houghton Library's collection of papyri
— The Ancient World Digital Library (AWDL)
— GRK-Papyri: A Dataset of Greek Handwriting on Papyri
— Trismegistos Texts Database
— papyri.info
— British Library Catalogues Archives and Manuscripts
— Greek Papyrus of the Heidelberg Papyrus collection
— Prosopography to Soknopaiu Nesos
— The Berlin Papyrus Database (BerlPap)
— Bavarian State Library
— American studies in papyrology
— The "Papyrus Portal"
— Arabic Papyrology database
— Epigraphische Datenbank Heidelberg
— Searchable Greek Inscriptions

# Annexe B

# Centralisation des tableaux

Pour faciliter la lecture en évitant les allers-retours entre tableaux situés dans différents chapitres, tous les tableaux du document sont regroupés dans cette Annexe.

## B.1 Chapitre 3

| Langues | Nombre d'images |
|---|---|
| Greek | 14890 |
| Coptic | 1140 |
| Arabic | 404 |
| Demotic | 163 |
| Latin | 138 |
| Fayumic_Coptic | 129 |
| Demotic.Greek | 63 |
| Coptic.Greek | 26 |
| Greek.Latin | 20 |
| Hieratic | 15 |
| old_Coptic | 12 |
| Arabic.Greek | 10 |
| Autres | 19 |
| **Total** | 17029 |

Tableau 3.1 – Nombre d'images par langues dans la collection de l'Université du Michigan. Lorsque plusieurs langages sont séparés par un point, cela signifie que le document contient plusieurs langues.

| | Aléatoire | | Sans Texte | | Avec Texte | |
|---|---|---|---|---|---|---|
| | Papy-S-Net | Koch | Papy-S-Net | Koch | Papy-S-Net | Koch |
| Vrai Pos. | 0.80 | 0.74 | 0.75 | 0.76 | **0.82** | 0.72 |
| Vrai Neg. | 0.92 | 0.88 | 0.82 | 0.87 | **0.94** | 0.86 |
| Faux Pos. | 0.09 | 0.12 | 0.08 | 0.13 | **0.06** | 0.14 |
| Faux Neg. | 0.20 | 0.26 | 0.25 | 0.24 | **0.18** | 0.28 |

TABLEAU 3.2 – Comparaison des performances sur la base *B500* obtenues par Papy-S-Net et Koch et. al. En gras le meilleur résultat pour chaque métrique, et en souligné le pire.

| | Hisfrag train | Hisfrag test | Michigan |
|---|---|---|---|
| Nb papyri | 17222 | 2732 | 1118 |
| Nb fragments | 101706 | 20019 | 4579 |
| mean frags/papy | 5.9 | 7.3 | 4.4 |
| std frags/papy | 9.7 | 2.9 | 6.4 |

TABLEAU 3.3 – Résumé des statistiques sur les bases *Michigan* et *Hisfrag*.

| | | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|
| Hisfrag | VGG16 | **0.67** | **0.87** | **0.75** | **0.92** |
| | Resnet50 | 0.61 | 0.83 | 0.71 | 0.82 |
| | Papy-S-Net | 0.52 | 0.75 | 0.6 | 0.86 |
| Michigan | VGG16 | 0.41 | 0.68 | 0.47 | 0.85 |
| | Resnet50 | **0.54** | 0.67 | **0.59** | **0.92** |
| | Papy-S-Net | 0.44 | **0.73** | 0.48 | 0.87 |

TABLEAU 3.4 – Comparaison des performances des trois architectures sur les deux bases de données. Ici, nous entraînons les modèles directement avec de l'information *niveau-document*. Les résultats sont calculés sur environ 800 fragments provenant du jeu de données Hisfrag, et environ 450 fragments provenant de la base de données Michigan.

## B.2 Chapitre 4

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | 0.21 | 0.37 | 0.29 | 0.59 |
| Resnet50 | **0.26** | **0.47** | **0.31** | **0.63** |

TABLEAU 4.1 – Évaluation des modèles pré-entraînés sur la base Hisfrag sans ré-entraînement. Évaluation sur le jeu de test de la base Michigan.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | **0.34** | **0.60** | **0.41** | **0.79** |
| Resnet50 | 0.31 | 0.56 | 0.37 | 0.72 |

Tableau 4.3 – Évaluation des modèles pré-entraînés sur la base Hisfrag et ré-entraînés avec des 50 papyrus de la base Michigan. Évaluation sur le jeu de test de la base Michigan.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | 0.07 | 0.07 | 0.06 | 0.29 |

Tableau 4.2 – Le modèle est entraîné avec seulement 50 papyrus de la base Michigan et évalué sur l'ensemble du jeu de test de la base Michigan.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| VGG16 | 0.35 | 0.62 | 0.42 | 0.81 |
| Resnet50 | **0.45** | **0.74** | **0.49** | **0.86** |

Tableau 4.4 – Évaluation des modèles pré-entraînés sur la base Hisfrag et ré-entraînés avec 1000 papyrus de la base Michigan. Évaluation sur le jeu de test de la base Michigan.

| | | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|
| Hisfrag | VGG16 | 0.38 | **0.73** | 0.47 | 0.72 |
| | Resnet50 | **0.41** | 0.51 | **0.48** | **0.78** |
| Michigan | VGG16 | **0.38** | **0.61** | **0.43** | **0.84** |
| | Resnet50 | 0.30 | 0.43 | 0.39 | 0.76 |

Tableau 4.5 – Evaluation des modèles entraînés en mode *auto-supervisé*. Les modèles ont été entraînés avec seulement de l'information niveau-fragment et évalués avec de l'information niveau-document sur 100 documents.

| | | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|
| Michigan | VGG16 | 0.32 | 0.32 | 0.38 | 0.78 |
| | Resnet50 | **0.43** | **0.75** | **0.47** | **0.84** |

Tableau 4.6 – Evaluation de modèles ré-entraînés en mode *auto-supervisé* à partir de modèles pré-entraînés sur Hisfrag.

| | documents | fragments | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|---|
| VGG16 | 25 | 213 | 0.38 | 0.61 | 0.43 | **0.90** |
| Resnet50 | 25 | 213 | 0.33 | 0.41 | 0.44 | 0.83 |
| VGG16 | 50 | 394 | **0.44** | **0.70** | **0.52** | 0.87 |
| Resnet50 | 50 | 394 | 0.27 | 0.40 | 0.36 | 0.71 |
| VGG16 | 75 | 651 | 0.23 | 0.40 | 0.29 | 0.63 |
| Resnet50 | 75 | 651 | 0.23 | 0.27 | 0.31 | 0.67 |
| VGG16 | 100 | 735 | 0.12 | 0.21 | 0.15 | 0.44 |
| Resnet50 | 100 | 735 | 0.19 | 0.25 | 0.27 | 0.56 |

TABLEAU 4.4.A – Cas d'utilisation réaliste avec des données de la base Michigan. Chaque modèle est entraîné en mode *auto-supervisé* et évalué sur les mêmes données qu'à l'entraînement. Ce tableau référence les données utilisées dans la Figure. 4.4.

| | documents | fragments | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|---|---|
| VGG16 | 25 | 542 | **0.63** | **0.75** | **0.66** | **0.96** |
| Resnet50 | 25 | 542 | 0.57 | 0.64 | 0.61 | 0.91 |
| VGG16 | 50 | 832 | 0.39 | 0.59 | 0.42 | 0.80 |
| Resnet50 | 50 | 832 | 0.50 | 0.57 | 0.58 | 0.86 |
| VGG16 | 75 | 1479 | 0.31 | 0.54 | 0.34 | 0.66 |
| Resnet50 | 75 | 1479 | 0.44 | 0.56 | 0.52 | 0.78 |
| VGG16 | 100 | 1012 | 0.37 | 0.64 | 0.44 | 0.70 |
| Resnet50 | 100 | 1012 | 0.51 | 0.62 | 0.60 | 0.81 |

TABLEAU 4.4.B – Cas d'utilisation réaliste avec des données de la base Hisfrag. Chaque modèle est entraîné en mode *auto-supervisé* et évalué sur les mêmes données qu'à l'entraînement. Ce tableau référence les données utilisées dans la Figure. 4.4.

| | mAP | top-1 | pr@10 | pr@100 |
|---|---|---|---|---|
| Baseline | 0.44 | 0.68 | 0.52 | 0.91 |
| Augment_hflip | 0.45 | 0.67 | 0.54 | 0.93 |
| Trigger_10 | 0.48 | 0.77 | 0.55 | 0.91 |
| Trigger_10_augment_hflip | **0.50** | **0.80** | **0.57** | **0.91** |
| Trigger_10_augment_hflip_vflip | 0.48 | 0.77 | 0.56 | 0.92 |

TABLEAU 4.8 – Comparaison des stratégies de sélection dans les batchs avec de l'augmentation de données. *Trigger_10* signifie que le processus de sélection dans les batchs a été déclenché à partir de l'epoch 10.

## B.3 Chapitre 5

| | |
|---|---|
| Première place | 6 |
| Seconde place | 2 |
| Troisième place | 0 |
| Top 10 | 12 |
| Top 20 | 12 |
| Top 50 | 19 |
| Top 100 | 24 |
| Total | 42 |

Tableau 5.1 – Synthèse des classements retournés par notre modèle pour les fragments dont on connaît les raccords existants.

| **Requête** | **Raccord** | **Classement** | **Requête** | **Raccord** | **Classement** |
|---|---|---|---|---|---|
| 1394 | 1370b | 168 | 0567i | 0567j | 5 |
| 1370b | 1394 | 161 | | 0567a | 164 |
| 1198 | 0811b | 34 | 0567j | 0567i | 6 |
| 0811b | 1198 | 23 | | 0567a | 198 |
| 1270 | 1256 | 1 | 0567a | 0567j | 229 |
| 1256 | 1270 | 1 | | 0567i | 253 |
| 563 | 0567k | 289 | 0812a | 0812b | 2 |
| 0567k | 563 | 313 | | 2275 | 5 |
| 1237a | 1237b | 48 | 2275 | 0812a | 1 |
| 1237b | 1237a | 42 | | 0812b | 2 |
| 2855d | 2855a | 50 | 0812b | 0812a | 1 |
| | 2855c | 101 | | 2275 | 4 |
| | 2855e | 298 | 1306e | 1306d | 33 |
| 2855e | 2855c | 67 | 1306d | 1306e | 29 |
| | 2855d | 199 | 0567e | 1189 | 224 |
| | 2855a | 203 | 1189 | 0567e | 139 |
| 2855a | 2855d | 53 | 796 | 0750b | 1 |
| | 2855c | 223 | 0750b | 796 | 1 |
| | 2855e | 330 | 1200 | 1238 | 99 |
| 2855c | 2855d | 63 | 1238 | 1200 | 139 |
| | 2855e | 73 | | | |
| | 2855a | 174 | | | |

Tableau 5.2 – Classements retournés par notre modèle sur les raccords de fragments donnés par les papyrologues (calculés sur les rectos).

# Table des matières