PROJET DE FIN D'ÉTUDES

Classification de papyrus anciens

Clients : Antoine Pirrone, Marie Beurton-Aimar et Nicholas Journet

Auteurs : Eric Barre, Benjamin Régnard, Thibaut Mazière

2020

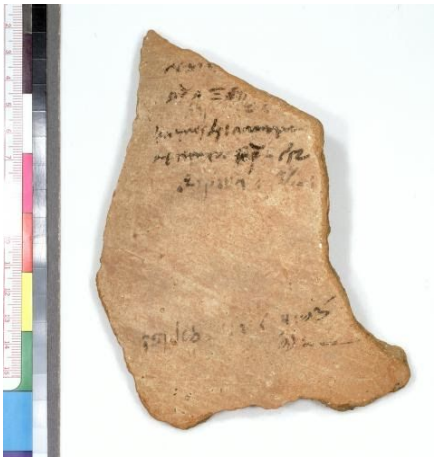
I) Introduction (contexte, problématique, sujet)

Pour notre projet de fin d'études nous avons choisi le sujet "*Classification de fragments de papyrus anciens*" proposé par Antoine Pirrone, Marie Beurton-Aimar et Nicholas Journet.

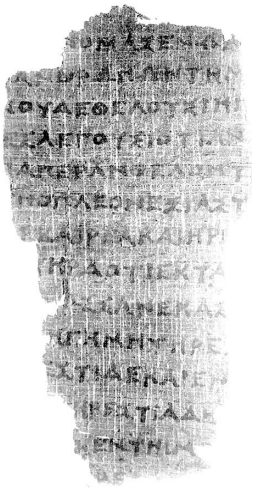
Ils disposent d'une collection de fragments de papyrus anciens qu'ils veulent rassembler pour reconstituer les papyrus d'origine.
Les papyrus pouvant dater d'époques très lointaines et faîtes de différentes façons à des endroits très différents. La diversité incroyable qui les compose peut aider à les rassembler mais la tâche reste cependant très dure.

Reconstitution d'un fragment de papyrus visible

Pour trouver les fragments appartenant au même papyrus, ils ont créé un modèle de réseau de neurones siamois, mais ils n'analysent que les images couleurs de ces fragments, or nous disposons aussi des images infra-rouges de ces mêmes fragments.



Fragment de papyrus "visible" [2]



Fragment de papyrus infrarouge [3]

Le but de notre projet est donc d'adapter ce réseau aux images infrarouges et de déterminer quel type d'image serait le plus pertinent ou si une combinaison des features des deux types permettrait d'améliorer la classification.

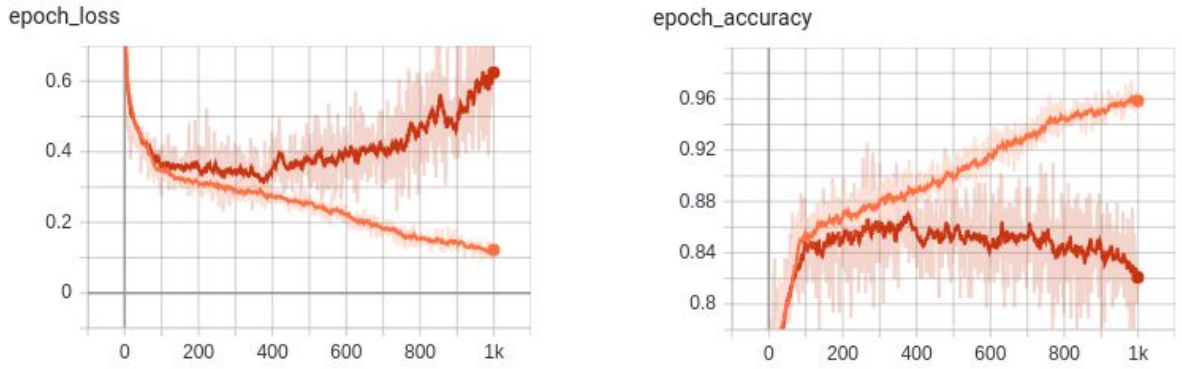
II) Etat de l'art / existant

Nos clients nous ont d'abord donné à lire leur article [1] consacré autour de la construction de leur réseau, comment il s'organise (voir partie VI.Architecture), et comment fonctionne la reconnaissance de fragments appartenant au même papyrus.

Le réseau, tout comme notre cerveau, essaye de reconnaître des patterns similaires entre différents patchs. Que ce soit au niveau de la graphie, de l'encre ou des qualités du support.

Nous avons donc dès le début réussi à bien comprendre le sujet et nous avons pu directement commencer à réfléchir à comment aborder le problème.

Ayant accès au dépôt github de notre client, nous disposons de plusieurs fichiers contenant des fonctions et modèles déjà fonctionnels.
En effet, dans le fichier *models.py*, un réseau de neurones siamois déjà fonctionnel sur les images couleurs des fragments de papyrus avait été créé et notre client l'avait déjà entraîné et nous avait fait part de ses observations :



Entraînement du réseau de neurones : à gauche la loss, à droite l'accuracy, en rouge foncé la courbe de validation et en rouge clair la courbe de train

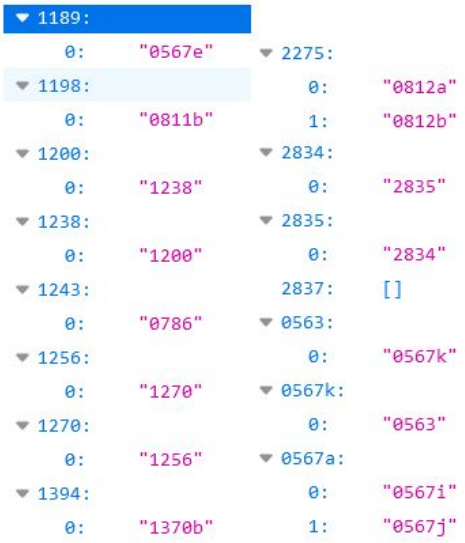
Nous pouvons voir que malgré l'overfitting nous obtenons un un bon score de 85% de précision.

Parmi les autres fichiers du client, nous nous sommes servis de :

- *make_dataset.py* permettant de pré-traiter les fragments de papyrus
- *train.py* permettant de lancer l'entraînement du réseau
- *infer_geshaem.py* permettant l'évaluation du modèle et générer un fichier csv des prédictions
- *generate_top_csv_from_csv.py* permettant de trier les prédictions par ordre décroissant

Nous avons aussi bien entendu accès à un dossier contenant les fragments de papyrus visible et un dossier contenant les fragments de papyrus infrarouges.
Nous avons 314 fragments de papyrus infrarouges "valides" c'est-à-dire qu'il forme une paire, nous possédons le même fragment en visible et en infrarouge.

Notre client nous a aussi fourni une vérité terrain d'environ 30 fragments (sous la forme d'un fichier JSON).
Grâce à cela, il sera possible pour nous aussi de calculer des métriques intéressantes et de comparer les prédictions du modèle du client avec les prédictions que l'on aura avec les fragments en infrarouge.



Extrait de la vérité terrain GT.json

Par exemple ici nous savons que le fragment numéro 1198 fait parti du même papyrus que l'échantillon n°0811b

Le client nous a également fourni son rapport, intitulé **Self-Supervised Deep Metric Learning for ancient papyrus fragments retrieval** [2], sur son réseau et son projet, qui nous a permis de de nous en imprégner.

Nous nous sommes également servis de l'article de l'article de Tooba Shams et Pascal Desbarats **Asian hornets nest's detection on drone acquired FLIR and color images using deep learning methods** [5].

Nous détaillerons plus tard comment mais cet article nous a éclairé sur la façon dont on peut fusionner des résultats d'entraînements de différents réseaux, et plus particulièrement un réseau qui traite des images visibles et un qui traite des images infrarouges.

III) User Stories

Quelles sont les fonctionnalités qu'un utilisateur pourrait lancer et pourquoi ?

Un utilisateur pourrait utiliser notre réseau si il veut :

- Lancer un entraînement sur des images visibles sans leurs fichiers annexes
- Lancer un entraînement sur des images visibles avec leurs fichiers annexes
- Lancer un entraînement sur des images IR
- Visualiser l'entraînement du réseau en direct ou après fin la fin de l'entraînement sur tensorboard
- Lancer une évaluation de son réseau entraîné
- Visualiser les prédictions de son réseau sur les fragments de papyrus de son choix
- Visualiser les différentes métriques liées à l'entraînement (rang moyen des fragments de la vérité terrain, matrice de confusion etc)

IV) Besoins du client

A) Fonctionnels

- Indiquer l'avancement de l'entraînement du réseau sur le terminal de l'utilisateur.
- Indiquer l'avancement de l'évaluation du modèle sur le terminal de l'utilisateur.
- Création de fichiers CSV permettant la visualisation des prédictions pour chaque fragments. L'utilisateur pourra ainsi rechercher facilement une prédiction d'une paire de fragments en particulier.
- Création d'un dossier regroupant les fragments les plus probables d'appartenir à chaque fragments pour que l'utilisateur puisse visualiser et analyser rapidement quels sont les fragments qui ont été associés.
- Création de scripts python facilement exécutables permettant à l'utilisateur de consulter différentes métriques et sauvegarde de la matrice de confusions en format JPG dans le dossier courant.

B) Non fonctionnels

Pour notre projet, étant un projet de deep learning, les principales contraintes sont des contraintes de temps. En effet, nous travaillons sur un nombre conséquent de données et l'entraînement du réseau ainsi que l'évaluation du modèle peuvent donc être très longs selon l'ordinateur dont vous disposez.
Si l'utilisateur ne possède pas un GPU assez puissant, il devra lancer les opérations à distance en SSH sur les ordinateurs du CREMI mais certains problèmes se posent :

- Il faut une connexion internet stable car si il y a coupure il y a risque que la connexion en SSH soit brisée et donc les opérations arrêtées.

Si l'utilisateur possède un matériel adéquat, le problème de la connexion internet ne se pose pas puisque les opérations se feront en local mais pour que tout marche il devrai au préalables :

- Avoir installé Python et les bibliothèques nécessaires au bon fonctionnement de nos programmes (voir partie "Choix logiciels argumentés")

Dans tous les cas l'utilisateur doit :

- disposer de la collection de fragments de papyrus visible et / ou IR

L'utilisateur doit pour accéder aux logs de l'entraînement du réseau, il faut :
- que le logiciel sauvegarde les logs à un endroit choisi par l'utilisateur (endroit temporaire ou non)

Durant l'entraînement il y a un risque d'overfitting, l'utilisateur va vouloir peut être évaluer le modèle à une époque spécifique où il n'y avait pas encore d'overfitting :

- sauvegarder le modèle toutes les X époques (X est un nombre choisi par l'utilisateur) sous forme de fichier .h5.

V) Choix logiciels (bibliothèques / langage)

Nous avons utilisé Python et Tensorflow car nous avons continué le travail du client qui avait déjà utilisé ces technologies. Ces technologies étant adaptées nous n'en avons pas changées.
L'utilisation de opencv est nécessaire pour le prétraitement des images.

VI) Architecture

Nous avons repris les scripts existants du client, qui permettaient d'entraîner un réseau sur des papyrus visibles. Nous avons donc adapté ce code à nos besoins pour qu'il fonctionne sur des images infrarouges de ces mêmes papyrus. Pour ce faire nous avons d'abord testé les algorithmes existants et avons vu qu'il y avait beaucoup de restrictions quant à l'organisation des fichiers dans les dossiers.

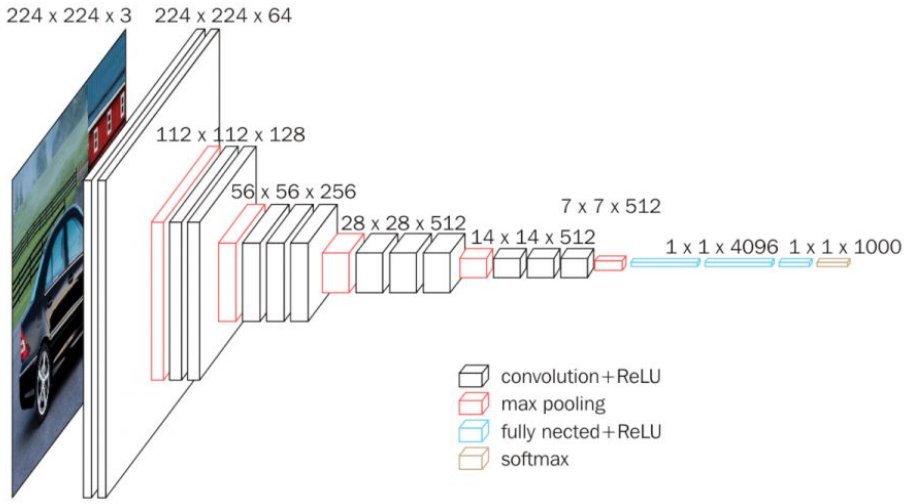
Nous avons donc créé un script qui nettoie les dossiers contenant les images pour n'avoir que les images utiles à l'entraînement qui restent dans lesdits dossiers.

```
ebarre002@scotty:~/espaces/travail/Papyrus/git2/IJDAR2020_code$ /net/ens/python/python3.8/bin/python3.8 difference.py
file : 2847_r_CL.JPG
file : 0258c_r_CL.JPG
file : 1322p_r_CL.JPG
file : 2888f_r_CL.JPG
file : 1321m_r_CL.JPG
file : 0088b_r_CL.JPG
infered not found in first folder :  0088b_r_CL.JPG
```

Une fois fait nous avons retouché les fichiers existants pour prendre en compte les fichiers infrarouges dans la préparation du dataset et l'entraînement. La préparation du dataset transforme les images en fichiers pickle pour l'entraînement.

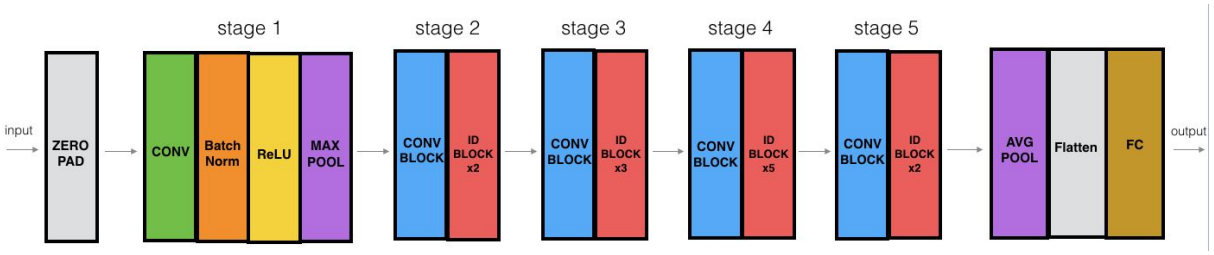
Le réseau est, comme souvent pour les réseaux siamois, un réseau convolutionnel.
Un réseau siamois est un réseau qui prend en entrée deux données différentes sur lesquelles travailler.
A partir des réseaux convolutionnels on va enlever les couches denses pour ne garder que la partie d'extraction des features. On va utiliser cette partie là de VGG16 ou de Resnet50 pour les réseaux plus profonds.

Le VGG16, composé de couches convolutionnelles successives est utilisé pour récupérer des features complexes.



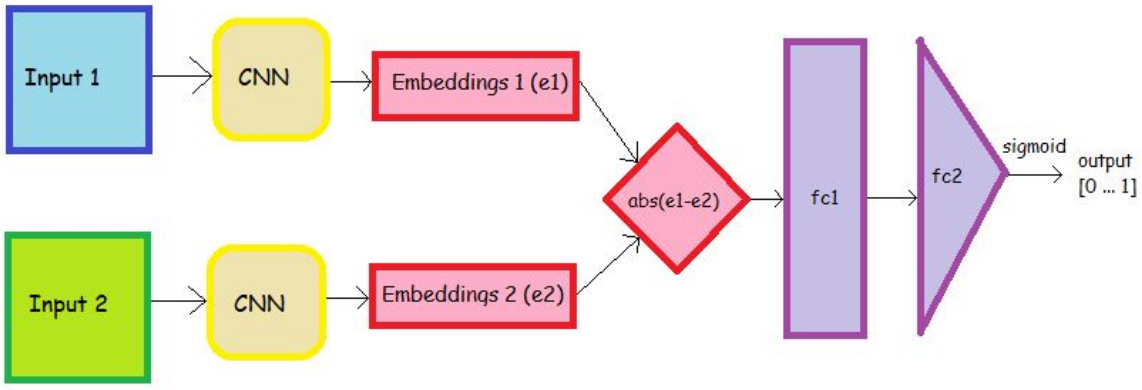
VGG16

Le Resnet50 a lui pour avantage d'avoir des "connections sautées" , ce qui diminue fortement les problèmes de gradients qui disparaissent après de nombreuses couches dans un réseau profond.



Resnet50

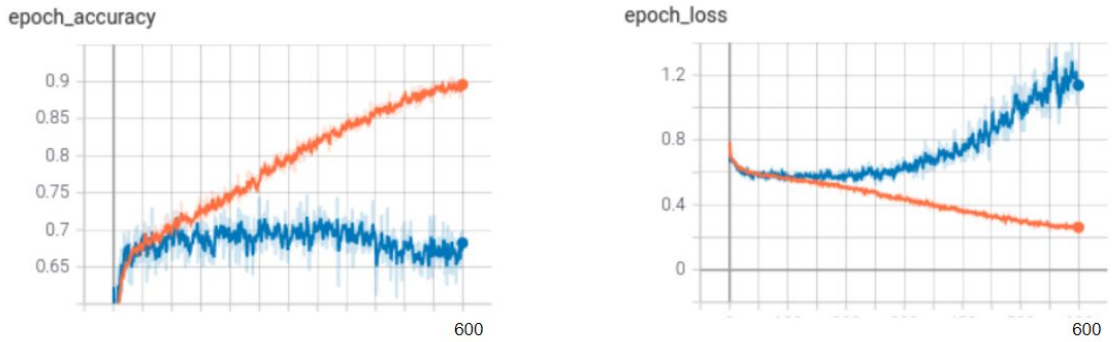
Schéma du réseau Siamois utilisé :



Réseau siamois

On peut voir qu'on extrait des features dans la partie jaune (CNN dans la figure et qu'ensuite on fait la difference des features extraites par les deux entrées. Cette différence est ensuite donnée à des couches denses.

Nous avons tout d'abord entraîné ce réseau sur les images du set Infrarouge:



Entraînement du réseau sur les images infrarouges : à gauche l'accuracy, à droite la loss, en bleu la courbe de validation et en rouge la courbe de train

Nous pouvons voir qu'actuellement il y a encore des problèmes d'overfitting alors que nous avons augmenté fortement le nombre de patchs que l'on présente au réseau.

A l'issue de l'entraînement avec des fragments visibles on obtient un taux de précision supérieur qu'après l'entraînement avec des fragments infrarouges. Ce qui est plutôt logique étant donné qu'on a uniquement des informations sur un canal de couleur. Nous obtenons donc aux alentours de 72% de précision avec les images infrarouges et 85% de précision avec les images visibles.

Lorsque l'on crée des pairs que l'on présente au réseau, si l'on crée toutes les paires possibles on aura beaucoup plus de pairs non similaires que similaires. Le script crée donc des mini patchs qui vont gérer cet équilibre.

Le réseau extrait plusieurs patchs de mêmes papyrus pour l'entraînement et essaye de n'en prendre que les parties les plus intéressantes à observer. Typiquement pour l'infrarouge on essaye de se focaliser sur l'écriture et pas sur la texture du papyrus.

VII) Réalisation

Notre client nous a fourni un modèle déjà fonctionnel et prêt à être entraîné sur des images visibles.
Nous avons donc tout d'abord dû modifier les fichiers que le client nous a donnés afin de pouvoir prendre en compte les images infrarouges.

A) Implémentation

1) Combinaison des prédictions des images visibles et infrarouges

Comme vu ci-dessus, nous avons entraîné nos modèles sur des fragments de papyrus visibles et infrarouge car le but de notre projet est de voir si une combinaison des features de ces deux types d'images est possible et si elle apporterait une meilleure précision finale.

Pour effectuer cette combinaison, nous nous sommes inspirés de l'article "Asian hornets nest's detection on drone acquired FLIR and color images using deep learning methods" écrit par Tooba Shams et Pascal Desbarats [5].
Dans cet article, les auteurs pondèrent les prédictions obtenues par le modèle entraîné sur les images visibles par les prédictions du modèle entraîné sur les images infrarouges par cette formule :

$$score = \alpha IR + (1 - \alpha)V$$

Avec α le coefficient de pondération, "IR" la prédiction de l'image IR courante par le modèle entraîné sur des images IR, et "V" la prédiction de l'image courante visible par le modèle entraîné sur les images visibles.
Nous déterminons si nous gardons ce nouveau score ou si nous laissons la prédiction visible de base sans pondération en fonction d'un certain seuil.
En effet, seules les prédictions inférieures à ce seuil seront affectées par cette pondération, ceci aura pour but d'éviter les faux positifs.

2) Implémentation dans notre projet

Pour appliquer cette formule à notre projet, reprenons nos deux modèles entraînés séparément sur des images visibles et infrarouges.

Nous allons d'abord générer les prédictions du modèle visible (grâce au script *infer_geshaem.py*) sur chacun de nos fragments, dans notre cas, une prédiction correspond au pourcentage de chances que le fragments x appartienne au même papyrus que le fragment y.

Un fichier .csv recensant toutes les prédictions pour toutes les paires de fragments va être généré, par exemple ci dessous, nous pouvons voir que le fragments 028c_r_CL a environ 42% de chances d'appartenir au même papyrus que la fragment 2847_r_CL.

	A	B
1		2847_r_CL
2	2847_r_CL	1
3	0258c_r_CL	0.42230746
4	1322p_r_CL	0.0011221224
5	2888f_r_CL	0.11185169
6	1321m_r_CL	0.008388895
7	0088b_r_CL	2.4857525e-05
8	1321k_r_CL	0.016431492
9	2907a_r_CL	0.36026996
10	2867b_r_CL	2.795372e-07
11	2853c_r_CL	0.6582234

Extrait du fichier predictions.csv

Nous allons ensuite appliquer la pondération de ces prédictions en fonction du seuil (ici nous avons pris 65% comme exemple) grâce à à la modification du script initial :

	A	B
1		2847_r_CL
2	2847_r_CL	1
3	0258c_r_CL	0.30414796229451896
4	1322p_r_CL	0.06372492227237672
5	2888f_r_CL	0.1284633748233318
6	1321m_r_CL	0.024552206043154
7	0088b_r_CL	0.055622633405982926
8	1321k_r_CL	0.0930290063843131
9	2907a_r_CL	0.3467470854520798
10	2867b_r_CL	0.04922701465972352
11	2853c_r_CL	0.6582228

Extrait du fichier predictionsPonderees.csv

Nous pouvons voir que pour notre paire de fragments 0258c et 2847 la prédiction est maintenant d'environ 30%, un changement significatif par rapport au 42% d'avant.
Nous remarquons aussi que pour la prédiction du fragment 2853c pour le 2847r n'a pas changée car elle est supérieure à 65%.

Nous obtenons donc deux fichiers csv distincts : un contenant les prédictions du modèle visible et un autre contenant ces prédictions pondérées par les prédictions du modèle infrarouge.

Nous allons créer pour chacun de ces deux tableaux un nouveau fichier *top.csv* qui va trier par ordre décroissant chaque fragments de papyrus par prédiction.
Grâce à ce tableau et à la vérité terrain fournie par notre client, nous allons pouvoir déterminer quelle méthode est la plus efficace, en effet nous pourrons retrouver les fragments de de papyrus de la vérité terrain dans ces fichiers top.csv et retourner les rangs de chaque fragments associés.
Ainsi nous pourrons voir si les rangs des fragments de la vérité terrain dans le top.csv des prédictions pondérés sont meilleurs que les rangs du top.csv des prédictions du modèle visible de base.

B) Résultats

Nous avons créé des scripts afin de comparer les résultats du modèle visible de base sur la vérité terrain avec ceux de la fusion des modèles visibles et infrarouges.
Les métriques que nous pensons intéressantes sont :

- Le rang moyen des fragments de la vérité terrain
- Une matrice de confusion
- La Mean Average Precision (mAP)

1) Rang moyen

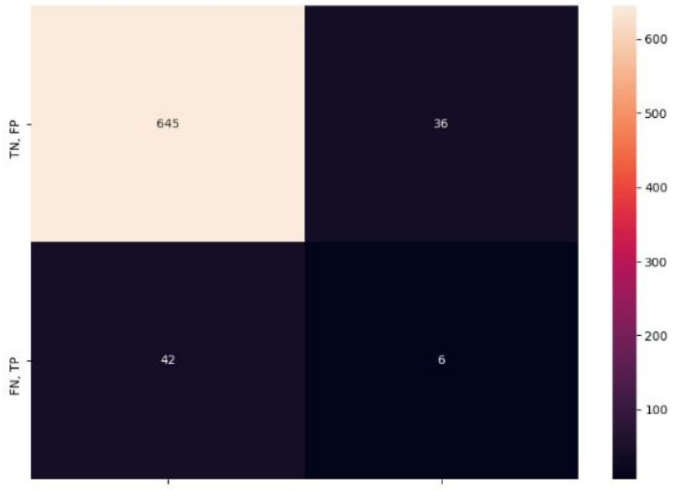
Faisons donc une comparaison entre les prédictions du modèle visible et les prédictions pondérées, tout d'abord le rang moyen des fragments de la vérité terrains :

- Avec les prédictions du modèle seulement entraîné sur des images visibles nous obtenons un rang moyen de 110/314 fragments
- Avec les prédictions visible pondérées par celles du modèle entraîné sur des images infrarouges nous obtenons un rang moyen de 124/314 fragments

La première observation que l'on peut faire est que le rang moyen est dans les deux cas pas suffisant, nous n'avons pas une assez bonne précision.
Nous savons aussi que la vérité terrain est très réduite (environ une trentaine de fragments) et donc il est difficile de tirer des conclusions sur seulement le rang moyen des fragments de cette vérité terrain.

2) Matrices de confusion

Analysons maintenant les matrices de confusions.
Tout d'abord choisissons un seuil où seulement les fragments ayant une prédiction supérieure à ce seuil sont considérés comme "vrais", appelons le le seuil de positivité.
Tout d'abord prenons un seuil de 65%.



Matrice de confusion de la vérité terrain avec prédictions du modèle visible

Matrice de confusion de la vérité terrain avec prédictions pondérées

Nous obtenons exactement la même matrice et c'est normal.
En effet nous n'avons appliqué la pondération seulement sur les prédictions inférieures à 65% or ici notre seuil de positivité était aussi de 65%.
De plus il y a très peu de prédictions supérieures à 65%, cela n'a donc pas vraiment d'impact.

Prenons maintenant un seuil de positivité de 50% :

Matrice de confusion de la vérité terrain avec prédictions du modèle visible

Matrice de confusion de la vérité terrain avec prédictions pondérées

Nous observons une réelle différence ici.
En effet, les faux positifs (de 104 à 64) ont beaucoup baissé, ce qui montre que la pondération par les prédictions des images IR a un réel impact, et comme prévu va permettre d'écarter certains faux positifs.
Mais nous observons une baisse des vrais positifs (de 12 à 6) qui sont donc passés dans les faux négatifs (de 36 à 42).

3) Mean Average Precision (mAP)

Nous avons calculé la" *mean average precision*" pour chacun des cas sur la vérité terrain, c'est la précision moyenne.
Nous avons utilisé l'outil sklearn.metrics.average_precision_score et avons obtenu :

- pour les prédictions du modèle visible : 0,0890482
- pour les prédictions pondérées : 0,0893278

Cette très faible mAP dans les deux cas est logique car le rang moyen est très bas.
La mAP pour les prédictions pondérées est très légèrement supérieure, ceci peut être dû à la forte baisse de faux positifs observée.

C) Conclusions autour des résultats

Avec nos différentes métriques calculées, nous pouvons en conclure que la pondération des prédictions du modèle visible par les prédictions a un réel impact, en effet la forte baisse des faux positifs est une réelle amélioration.
Mais la baisse des vrais positifs et un rang moyen de la vérité terrain plus grand (et donc moins précis) montrent qu'il y a aussi toujours des imprécisions.
Notre plus gros problème est que la vérité terrain est beaucoup trop petite, la baisse des vrais positifs est donc peut-être dû à certains cas particuliers de la vérité terrain.

VIII) Tests

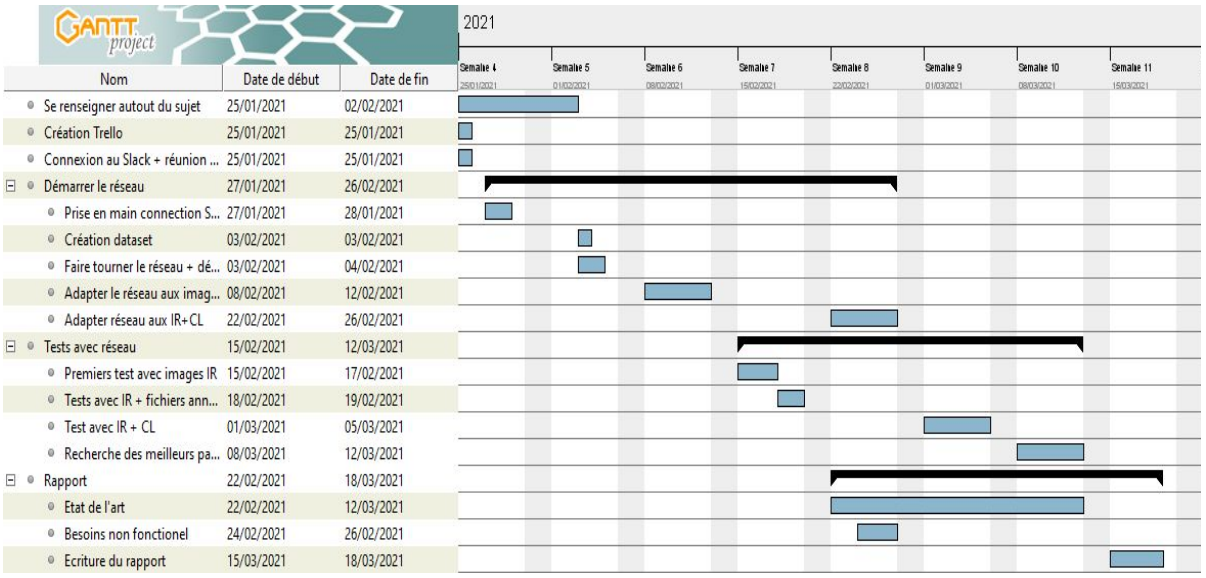
1) Tests de validation

Comme nous avons travaillé sur un projet déjà existant, il a fallu nous assurer que l'implémentation des nouvelles fonctionnalités n'allait pas altérer celles que le clients avait déjà mis en place et que nos nouvelles fonctionnalités marchent bien.

Nous avons testé :

- Qu'un entraînement des images visibles est toujours possible même après modification pour permettre l'entraînement des images infrarouges.
- Que seules les prédictions inférieures au seuil choisi étaient pondérées par les prédictions du modèle infrarouge.
- Que les métriques étaient bien calculées et cohérentes.

IX) Gantt Prévisionnel



Nom	Date de début	Date de fin
Se renseigner autout du sujet	25/01/2021	02/02/2021
Création Trello	25/01/2021	25/01/2021
Connexion au Slack + réunion ...	25/01/2021	25/01/2021
Démarrer le réseau	27/01/2021	26/02/2021
Prise en main connection S...	27/01/2021	28/01/2021
Création dataset	03/02/2021	03/02/2021
Faire tourner le réseau + dé...	03/02/2021	04/02/2021
Adapter le réseau aux imag...	08/02/2021	12/02/2021
Adapter réseau aux IR+CL	22/02/2021	26/02/2021
Tests avec réseau	15/02/2021	12/03/2021
Premiers test avec images IR	15/02/2021	17/02/2021
Tests avec IR + fichiers ann...	18/02/2021	19/02/2021
Test avec IR + CL	01/03/2021	05/03/2021
Recherche des meilleurs pa...	08/03/2021	12/03/2021
Rapport	22/02/2021	18/03/2021
Etat de l'art	22/02/2021	12/03/2021
Besoins non fonctionel	24/02/2021	26/02/2021
Ecriture du rapport	15/03/2021	18/03/2021

Gantt prévisionnel

X) Comparaison Gantt Prévisionnel / Effectif

Nous avons été un peu trop optimistes sur notre Gantt prévisionnel.
En effet, nous avions prévu une semaine de libre à la fin consacrée seulement à l'écriture du rapport en ayant fini toutes les tâches de tests autour du projet mais ça ne s'est pas passé comme cela dans la réalité.
En effet, durant cette dernière semaine nous nous sommes répartis des tâches de tests et de création de scripts permettant de mesurer différentes métriques par rapport à l'évaluation du modèle et nous avons aussi dû rédiger le rapport en même temps.
L'écriture du rapport a aussi commencé un peu plus tard que prévu.

Nous avons aussi sous-estimé les sous-tâches autour des tests du réseau et de l'évaluation du réseau. En effet, ce sont des tests qui prennent du temps à s'effectuer (parfois plusieurs heures) et il y avait plus de choses à tester que ce que nous pensions au départ.

XI) Conclusion

Durant ce projet, nous avons donc réussi à remplir les demandes du clients en complétant le projet qu'il avait déjà commencé.
Nous avons apporté un début de réponse quant à l'utilité de la fusion des features des images visibles et des images infrarouges, mais avec plus de temps, de tests, de données et une vérité terrain plus importante nous pourrons avoir une véritable réponse.

Nous avons donc obtenu des résultats mais pas forcément aussi bons que nous l'attendions. Mais ce projet nous a permis de mettre en place véritablement un système de travail de groupe avec une répartition claire et un découpage des tâches précis pour essayer de travailler efficacement.

Avoir un travail donné par un réel client que l'on voit toutes les semaines est très stimulant et nous rapproche d'une véritable expérience professionnelle.

Cette expérience nous a permis de grandement renforcer notre capacité de travail à distance et de travail sur des machines distantes, une compétence très intéressante à l'heure actuelle et sans doute de plus en plus importante au cours de notre carrière du fait de la situation sanitaire que nous connaissons tous.

Références

[1] Antoine Pirrone, Marie Beurton-Aimar et Nicholas Journet, *Papy-S-Net : A Siamese Network to match papyrus fragments* 18 Novembre 2019

[2] Advanced Papyrological Information System UM, *https://quod.lib.umich.edu/cgi/i/image/image-idx?c=apis&page=search*

[3] Excavating and Conserving Europe's Oldest Books: A Papyrus from Mangalia on the Black Sea (*P. Callatis 1*)

[4] Antoine Pirrone, Marie Beurton-Aimar et Nicholas Journet, Self-Supervised Deep Metric Learning for ancient papyrus fragments retrieval

[5] Tooba Shams, Pascal Desbarats, *Asian hornets nest's detection on drone acquired FLIR and color images using deep learning methods*

The research leading to this results has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under grant agreement No 758907 and is part of the GESHAEM project, hosted by the Ausonius Institute.

European Research Council

Established by the European Commission