Projet VEgA

Université Paul Valéry - Montpellier 3
Département Mathématiques et Informatiques Appliquées aux Sciences Humaines et Sociales
Mai 2020

Étudiants :
BRUN Fleur
FRANC Priscille
NOVALES Gaëlle
ORIOL Théo
RAMIREZ Sarah

Encadrants :
Commanditaire : BISTON-MOULIN Sébastien
Tuteur universitaire : SERVAJEAN Maximilien

Remerciements

Nous tenons à remercier particulièrement toutes les personnes nous ayant permis de réaliser ce projet qui fut très enrichissant et plein d'intérêt. Nous souhaitons donc dire merci à :

- **Sébastien Biston-Moulin,** commanditaire du projet, pour nous avoir permis de travailler sur ce sujet passionnant et avoir pris le temps de nous recevoir et de nous conseiller.
- **Maximilien Servajean,** tuteur universitaire, qui a su nous faire part de précieux conseils et répondre à nos questions tout au long de l'année.
- **Sandra Bringay et Sophie Lèbre,** responsables du master MIASHS, grâce à qui nous avons pu réaliser ce travail de groupe.

Table des matières

Introduction

Aujourd'hui, divers enjeux peuvent être identifiés dans la recherche en égyptologie. Tout d'abord, les données disponibles sur les hiéroglyphes sont relativement pauvres et parfois obsolètes. De plus, les chercheurs en égyptologie ne disposent d'aucun outil moderne de partage et de reconnaissance automatique de hiéroglyphes fiable. Ce manque de ressources et de technologie induisent une réelle perte de temps pour les chercheurs. Deux projets, "VEgA" et "Karnak", sont donc menes en cooperation entre le CNRS, l'Universite Paul Valery et le Ministere de la Culture et de la Communication. Notre travail s'inscrit dans le projet Karnak, qui consiste à recenser et editer toutes les inscriptions presentes sur les differents monuments du temple de Karnak. Cette demarche a pour but d'aider les egyptologues dans leur travail de recherche et leur permet de disposer d'une base de donnees de tous les monuments et signes presents dans le temple sous la forme d'une interface web. C'est dans ce contexte que notre projet intervient.

Nous avons pu définir, en accord avec le commanditaire, deux principaux objectifs à atteindre et en déduire le périmètre de notre projet. Dans un premier temps, nous avons établi la nécessité de réaliser un modèle de classification et de recherche de similarités afin d'être en capacité d'identifier, à partir d'une image de hiéroglyphe, les images similaires. L'objectif de cet outil est de pouvoir visualiser un hiéroglyphe dans les différents contextes dans lesquels il peut apparaître. Dans un second temps a émergé l'objectif de pouvoir détecter (et reconnaître) rapidement les différents hiéroglyphes présents sur une plaque entière. Ainsi, il serait possible de les indexer rapidement. Pour cela, il nous fallait créer un outil de détection des hiéroglyphes. Ces fonctionnalités étant destinées à être utilisées par des experts, il a été convenu qu'elles devraient être disponibles sur une interface web dont l'accès serait réservée aux experts. Nous venons donc de voir les différents objectifs incombant à notre projet ainsi que le périmètre des outils à réaliser pour y répondre. Pour cela, nous disposions d'une année à raison d'un total de 140 heures par étudiant. Ce projet, encadre par Sebastien Biston-Moulin et Maximilien Servajean, a ete realise par Fleur Brun, Priscille Franc, Gaelle Novales, Theo Oriol et Sarah Ramirez.

Ce document a pour but de presenter les differentes etapes ayant ete mises en place par le groupe Pixoglyphe dans le but de creer les outils précédemment cités. Il se présente de la manière suivante : Après avoir décrit les sources de données utilisées et les traitements effectués pour en augmenter la quantité, nous présentons et justifions en deux temps les différents modèles ayant été mis en place pour répondre à la problématique. Ensuite, nous abordons la question du site web, que ce soit du côté utilisateur ou développeur. Et enfin, nous détaillons notre mode de gestion de projet sous la forme d'une autocritique et d'une mise en avant des améliorations mises en place.

Chapitre 1

Base de données

La première étape de notre travail a été de constituer une base de données composée d'images de hiéroglyphes. Il est évident que pour pouvoir développer des modèles de deep learning fiables et performants, nous devions constituer une base de données bien réfléchie et particulièrement conséquente. Pour commencer, à la demande de Sebastien Biston-Moulin, nous avons alimenté la base de données d'images provenant directement du site officiel du projet Karnak et classées/labellisées par code de hiéroglyphe. Cependant, c'est un travail coûteux en temps. Afin d'améliorer notre base de données et d'éviter notamment le problème du sur-apprentissage, nous avions donc besoin d'effectuer un travail d'augmentation des données autrement que par collecte manuelle. Les différentes étapes de constitution de la base de données servant à la création du premier outil (i.e., de classification et de recherche d'images similaires) sont décrites dans cette partie.

1.1 Collecte d'images du site Karnak

Au cours de la première phase de cette création de base de données, diverses questions ont émergé, tant au niveau technique que du besoin réel du commanditaire :

- Que souhaite-t-on détecter ? Un signe, un mot ? Quelle est la différence ?
- Comment découper et structurer ces images ? Avec quel contenu, quel format ?
- Comment nommer les signes et les images ?
- Quelles images enregistrer ?

Source de nombreux débats, ce questionnement a également engendré un remaniement voire une perte du travail déjà effectué afin de rendre la base de données initiale la plus fiable et pertinente possible. Après réflexion et discussions avec l'expert métier Sébastien Biston-Moulin, nous avons décidé d'appliquer les conditions suivantes :

- L'image doit cibler un signe unique
- L'image doit être sous la forme carrée
- Le hiéroglyphe ciblé doit être centré
- L'image doit contenir du bruit autour du signe ciblé (afin que le modèle soit capable par la suite d'identifier un signe quel que soit le bruit qui l'entoure)

Nous avons alors constitué une base de données contenant dix hiéroglyphes différents, avec entre 65 et 157 images par signe :

FIGURE 1.1 – Hiéroglyphes présents dans la base de données

De plus, pour chaque hiéroglyphe, nous devions essayer d'avoir des images provenant de plusieurs époques, auteurs, monuments (au sein du site Karnak), et même d'images de plus ou moins bonne qualité afin d'entraîner un modèle le plus généralisable possible. C'est ce que nous pouvons voir illustré ci-dessous :

FIGURE 1.2 – Différences de qualité des images de la base de données

Par manque d'expertise sur les hiéroglyphes, nous avons fait appel à Sébastien Biston-Moulin pour vérifier chaque image et reclasser celles pour lesquelles nous avions fait erreur.

Au format png ou jpg, les images sont importées directement grâce à python à partir d'une structure sous forme d'arborescence afin de différencier les labels des 10 hiéroglyphes. En étant importée, elles sont également retaillées à un format identique de 224*224 pixels.

Ce premier travail effectué présente malgré tout plusieurs limites. En effet, nous nous intéressons ici seulement au site Karnak, ce qui restreint forcément la provenance des signes tant au niveau des monuments que de l'époque de gravure. De plus, notre base de données permet d'entraîner un modèle identifiant seulement dix hiéroglyphes différents alors qu'il en existe des milliers.
Enfin, le nombre d'images récoltées par signe est relativement faible pour une exploitation en deep learning. Nous avons donc décidé de mettre en place différentes techniques pour augmenter notre jeu de données et améliorer les performances du modèle de classification : la data augmentation et le web scraping. Pour cela, nous nous sommes notamment appuyés sur deux articles du site web Nanonets : How to use Deep Learning when you have Limited Data - Part 1 (Sarthak Jain, 2017) et – Part 2 (Arun Gandhi, 2019).

1.2 Data augmentation

La 'data augmentation', comme son nom l'indique, consiste à accroître la quantité de données sur lesquelles entraîner le modèle choisi mais aussi à en augmenter la diversité sans avoir à collecter manuellement des données. Il s'agit simplement d'effectuer des modifications sur les données déjà récoltées. Ainsi, le modèle de classification sera plus 'robuste', c'est-à-dire qu'il sera capable de classer correctement les images quelles que soient les conditions de prise de l'image. Parmi celles-ci, on peut notamment citer les facteurs de zoom, de luminosité, de point de vue ou encore de positionnement de l'objet d'intérêt dans l'image. Ces conditions seront donc artificiellement recréées grâce aux techniques de data augmentation.

Nous avons choisi d'appliquer au jeu de données servant à la classification six des transformations les plus courantes grâce à la fonction transforms () de la librairie Torchvision :

- Le décalage horizontal, comblant automatiquement le vide ainsi créé à gauche ou à droite de l'image ;
- Le décalage vertical, comblant automatiquement le vide ainsi créé en haut ou en bas de l'image ;
- La rotation aléatoire, qui consiste en une simple rotation de l'image selon un angle aléatoire (ici variant de -15 à 15 degrés) ;
- L'inversion horizontale, ou effet miroir ;
- La variation des paramètres de l'image à travers la modification aléatoire de la luminosité, du contraste et/ou de la saturation ;
- Le recadrage ou zoom aléatoire, qui consiste à sélectionner une partie (ici 80 à 100%) de l'image de départ et à redimensionner cette partie à la taille souhaitée.

Nous avons ainsi multiplié par six la quantité d'images fournies au modèle. Il aurait bien sûr été possible d'appliquer d'autres modifications, tout aussi simples ou plus complexes (e.g., l'effacement aléatoire d'une zone de l'image). Toutefois, les transformations déjà effectuées semblaient suffisantes pour obtenir de bonnes performances et il n'est pas nécessaire de pousser la diversité à des conditions qui ne pourraient réellement se produire. Bien sûr, il se peut qu'une partie du hiéroglyphe soit effacée donc il aurait pu être intéressant d'ajouter un effacement aléatoire d'une zone de l'image, mais cette condition est déjà présente dans notre base de données de départ.

1.3 Web scraping et transfer learning

Nous souhaitions également pré entraîner le modèle sur un grand nombre d'images de hiéroglyphes non issues de notre base de données. En théorie, plus le nombre d'images sur lequel le modèle est entraîné est grand et fidèle/similaire aux images que l'on souhaite classifier, plus le résultat sera précis. Notre objectif était donc d'essayer d'améliorer notre modèle de classification d'images en entrainant une première fois le modèle (déjà pré entraîné sur d'autres données) avec des images de hiéroglyphes issues du web, puis d'effectuer du 'transfer learning'. Cette technique consiste à utiliser un modèle pré-entraîné sur autre jeu de données sur nos propres images, bénéficiant ainsi de la connaissance acquise sur le premier jeu de données pour mieux traiter le nôtre. Le modèle serait alors pré-entraîné avec des images proches de celles que l'on souhaite classifier avant d'être entraîné sur notre jeu de données.

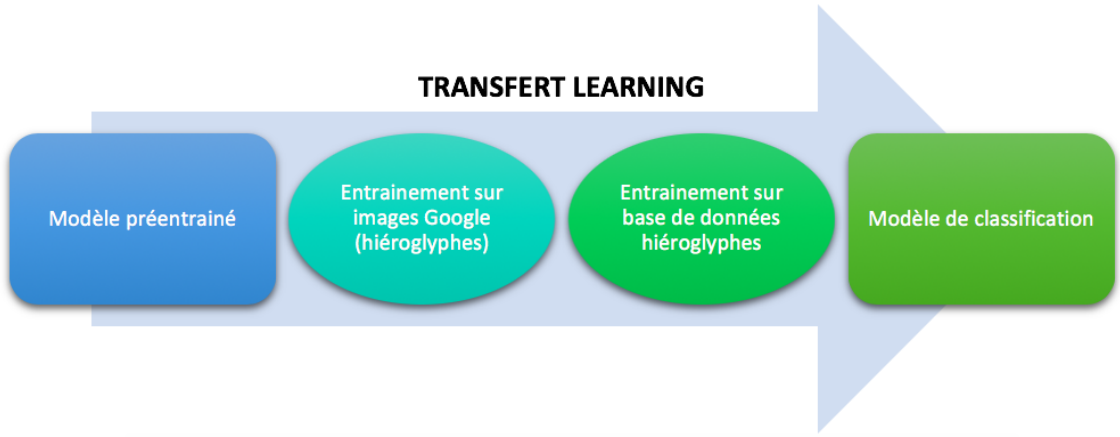

FIGURE 1.3 – Pipeline du 'transfert learning'

Afin de collecter un grand nombre d'images de hiéroglyphes, nous disposons d'une base de données accessible via le web : Google image. Grâce à cela, nous pouvons effectuer la recherche du mot « hiéroglyphe », couplé à des filtres de couleur, et obtenir une majorité d'images de hiéroglyphes. La plupart sont en réalité des images contenant un ensemble de hiéroglyphes sur une même plaque. Certaines ne correspondent pas du tout à des images de hiéroglyphes. Cependant, celles-ci étant plutôt rare, ce n'est pas un problème ? Au contraire, elles pourront avoir une fonction de biais lors de l'apprentissage.

Afin de récolter un maximum d'image en peu de temps, l'automatisation de la collecte était indispensable ; c'est pourquoi nous avons choisi la méthode du « web scraping ». Pour cela, nous avons utilisé python afin d'uniformiser les langages de programmation utilisés et de pouvoir utiliser le GPU de l'université. Cependant, il s'est avéré que Google bloque le webscraping possible à un maximum d'une centaine d'images pour un URL Google image. De plus, le chargement des images était effectué de manière très aléatoire, indépendamment de notre volonté. Ce projet n'a donc finalement pas abouti. Il aurait probablement aidé à l'amélioration des performances de notre modèle et mériterait sans doute d'être développé. Cependant, nous avons opté pour une autre technique, que nous développerons un peu plus loin dans ce rapport : l'active learning. Cela devrait également permettre d'augmenter progressivement mais significativement la base de données et de pallier le manque d'images sur lequel le modèle de classification est entraîné. Voyons maintenant la mise en place de ce dernier.

Chapitre 2

Objectif 1 : Classification et recherche de similarités

Pour répondre à la problématique du commanditaire, nous avons tout d'abord essayé d'utiliser une ACP (Analyse par composantes principales) puis un KNN (K-nearest neighbors). Cependant, il est vite apparu qu'il ne serait pas possible d'utiliser un modèle aussi simple pour trouver les images les plus proches. En effet, il est nécessaire de prédire au préalable la classe d'un hiéroglyphe et de représenter son image dans un nouvel espace vectoriel pour être en mesure de trouver des images qui soient réellement similaires en termes de sémantique. Nous avons donc dû commencer par mettre en place un modèle de classification d'images. Avant d'entrer dans le détail des modèles, la pipeline de ce qui était prévu est présentée ci-dessous :

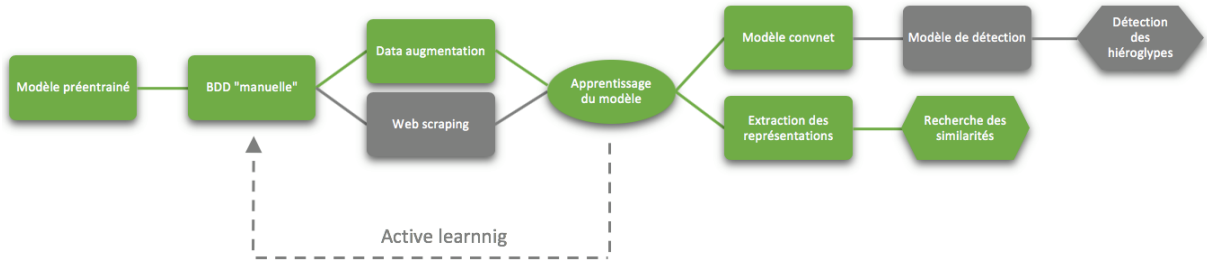

FIGURE 2.1 – Pipeline globale présentant les deux finalités

2.1 Modèle de classification

Pour le modèle de classification, nous avons commencé par un état de l'art et décidé d'utiliser un réseau de neurones convolutifs (ou ConvNet). L'intérêt principal de ce type de modèle pour nous est qu'il apprend automatiquement les caractéristiques de l'image. Concernant son fonctionnement, il est composé de 4 couches distinctes :

- **La couche de convolution,** qui extrait les caractéristiques de l'image reçue en entrée.
- **La couche de 'pooling',** qui permet de réduire la taille de l'image tout en gardant ces caractéristiques importantes.

- **La couche de non linéarité,** indispensable pour résoudre des problèmes complexes, permettant également de rendre le modèle plus rapide sans perdre en qualité.
- **La couche 'fully connected',** qui prend un vecteur en entrée et produit un vecteur de probabilité (de taille N hiéroglyphes ici) permettant la classification en sortie. Chaque nœud de cette couche est connecté à tous les neurones de la couche précédente.

Pour réaliser le modèle nous avons dans un premier temps utilisé Keras mais ce framework agit comme une boîte noir. Pour pouvoir comprendre et mieux paramétrer notre modèle, nous avons donc choisi de migrer entièrement vers un framework permettant de faire du deep learning de façon plus professionelle : Pytorch.

Afin d'améliorer les performances de notre modèle et le rendre le plus fiable possible, nous avons réalisé du 'transfer learning', présenté précédemment. Il existe déjà différents modèles pré-entrainés applicables à un cas de classification d'image, tels que VGG16 que nous avons décidé d'utiliser après avoir testé les performances de plusieurs modèles. Le modèle VGG16 est un Réseau Neuronal Convolutif entraîné sur plusieurs millions d'images de 1000 catégories différentes. Pour l'appliquer à notre set de données, nous avons extrait ses couches de convolution et ajouté nos couches fully-connected afin d'entraîner le modèle sur nos propre classes. Il renvoie ensuite le résultat de la dernière couche de la partie fully connected sur laquelle nous avons également appliqué une couche softmax pour obtenir des probabilités et connaître la classe prédite.

Si nous obtenions seulement 40% d'accuracy avant d'employer l'apprentissage par transfert, celui-ci nous a permis d'atteindre ensuite 80% d'accuracy. Bien sûr, les résultats seraient encore meilleurs avec davantage de données. En effet, pour le moment, le modèle a des difficultés à discerner efficacement des signes très similaires. Par exemple, seuls l'oeil d'un expert dissocie efficacement les deux signes ci-dessous :

FIGURE 2.2 – Deux signes différents

Pour que le modèle fasse la différence entre ces différents cercles, il aurait fallu davantage de connaissance donc de données.

2.2 Modèle de recherche des similarités

Le modèle de classification ayant permis de séparer les différents hiéroglyphes dans un nouvel espace vectoriel, nous nous sommes penchés sur l'objectif qui était de pouvoir renvoyer des images similaires (en termes de contenu hiéroglyphique) à celle fournie en entrée.

Nous avons commencé par extraire le vecteur de représentation de chaque image en supprimant la dernière couche de notre modèle qui transforme cette représentation en probabilité de label. La même procédure est appliquée à l'image en entrée : classification et récupération d'un vecteur de représentation. Une fois la représentation de chaque image obtenue, il existe plusieurs algorithmes de recherche des images les plus proches. En particulier, deux types de méthodes d'approximation sont utilisées : soit du hachage, soit des arbres.

Nous avons dans un premier temps choisi de tester la méthode des arbres avec l'utilisation du KD-tree, un algorithme de partitionnement spatial de l'espace à k dimensions permettant de structurer les données en fonction de leur répartition dans l'espace. Son intérêt est sa capacité à trouver les voisins les plus proches rapidement en évitant de comparer la nouvelle image à toutes les images contenues dans notre base de données.

Nous nous sommes également renseignés sur les méthodes dites de hachage qui consistent à transformer chaque image en un code. Une marge d'erreur est définie sur la différences que deux codes peuvent avoir pour être considérés comme similaires. C'est ainsi que deux images similaires partageront le même code. Cette méthode, particulièrement rapide, nécessite toutefois de déterminer différents paramètres. Le LSH, contrairement à d'autres méthodes de hachage, semblait intéressant dans notre cas car il permet d'identifier comme similaire des images même si elles sont légèrement différentes. Le risque existait malgré tout que cela nous renvoie des images de hiéroglyphes dans les mêmes contextes et pas dans des contextes différents car les méthodes de hachage restent sensibles aux distorsions. Or, tout l'intérêt de cette recherche de similarités résidait dans le fait que nous souhaitions disposer de différentes images d'un hiéroglyphe dans différents contextes. Ayant obtenu de bons résultats avec le KDTree et sachant cet inconvénient de la méthode LSH, nous avons décidé de conserver la première méthode.

2.3 Active learning

En parallèle à la recherche de similarités et afin d'améliorer les performances de notre modèle de classification, nous avons décidé d'utiliser l'active learning, ou apprentissage actif. Il s'agit là d'utiliser les images de hiéroglyphes importées par l'utilisateur, afin d'augmenter notre jeu de données. Cette technique intervient donc en sortie du modèle de classification.

Rappelons que celui-ci, pour chaque photo analysée, calcule la probabilité que l'image appartienne à telle ou telle classe. Dans notre cas, nous avons 10 classes (10 hiéroglyphes différents) donc 10 probabilités. À partir de ces probabilités, nous avons appliqué un théorème mathématique, l'entropie de shannon. Celle-ci permet de savoir si la probabilité maximum appartenant à la classe i, obtenue par le modèle, est fiable ou non. Contrairement aux probabilités en sortie du modèle, l'entropie de Shannon prend en compte le nombre de classes dont on dispose. Cela nous permet d'évaluer la probabilité d'appartenance à une classe d'un hiéroglyphe de façon objective indépendamment du nombre de classes. En effet, sur un ensemble de dix classes, une prédiction certifiant à 40% que le hiéroglyphe appartient à une classe n'a pas la même signification que sur un nombre total de mille classes par exemple. Dans le premier cas, la prédiction pourrait être jugée d'assez peu fiable tandis que dans le second, elle semblera plutôt très bonne. Pour l'utilisation de l'entropie de Shannon, nous avons défini un seuil fixe que nous avons établi à 80% de l'entropie maximum.

Pour une source x (image x d'un hiéroglyphe) comportant n classes, chaque classe xi ayant une probabilité Pi , l'entropie H de la source x est définie comme :

$$H(x) = -\sum_{i=1}^{n} P_i * ln(P_i)$$

Avec :

$$n = 10$$

$$ln(n) = entropiemax$$

Le seuil Y que nous avons définit se calculera, lui, de la façon suivante :

$$Y = (seuil) * ln(n)$$

Avec, dans notre cas :

$$seuil = 0.8$$

Selon l'entropie de Shannon H(X) obtenue, nous avons définir deux possibilités :

- **La classification si H(X) $>$ Y :** On peut considérer ici que la probabilité attribuée à la classe xi est fiable donc l'image est automatiquement labellisée et ajoutée dans la base de données.
- **La labellisation si H(X) $\leq Y$:** La classification de l'image n'étant pas fiable, elle est stockée dans un dossier avec l'entropie associée. Les experts pourront alors labelliser manuellement les images de ce dossier sur une page dédiée du site web. L'image affichée par défaut sur la page correspondra à l'image dont l'entropie est la plus faible. L'expert doit simplement renseigner le code du hiéroglyphe affiché et valider sa saisie. Une fonction python a été réalisée afin de récupérer le chemin de la photo et le code inscrit par l'utilisateur afin de placer/classer correctement l'image dans le jeu de données.

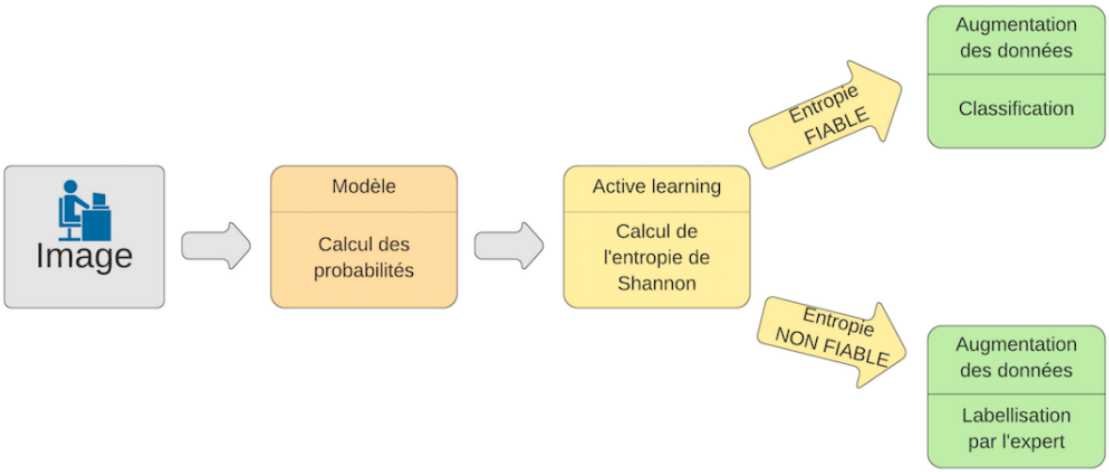

FIGURE 2.3 – Pipeline de l'active learning

Chapitre 3

Objectif 2 : détection de hiéroglyphes

Pour aller plus loin, nous avons commencé la réalisation du modèle de détection d'objet permettant d'identifier différents hiéroglyphes présents sur une image contenant plusieurs signes et non seulement un hiéroglyphe central. Pour cela, nous avons une fois encore commencé par faire un état de l'art dans le domaine et avons finalement choisi le modèle Faster Rcnn, ayant les meilleurs résultats en terme d'accuracy. Ce dernier point était en effet d'autant plus important que nous travaillons avec des images complexes : de très grande taille, avec des hiéroglyphes souvent en mauvais état. Nous verrons dans cette partie le fonctionnement de ce type de modèle, certains prérequis techniques propres à notre cas et enfin les perspectives d'évolution de cet outil. Nous avons également choisi de le réaliser avec pytorch.

3.1 Modèle de détection d'objets : fonctionnement

Le modèle de détection d'objets Faster Rcnn est divisé en quatre parties distinctes.

La première partie, ou backbone, est composé de deux modèles à convolution qui fournissent chacun en sortie une feature map.

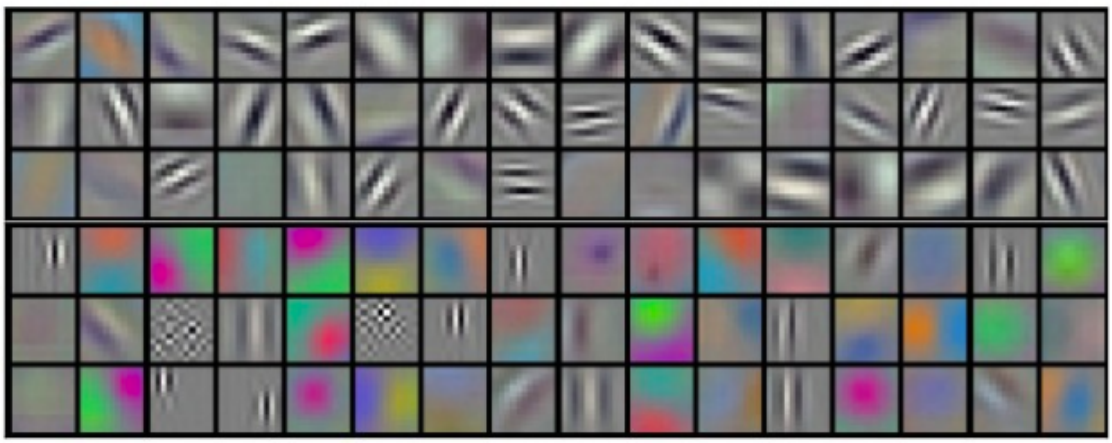

FIGURE 3.1 – Feature map

La deuxième partie ou 'Region proposal network' (RPN), prend en entrée la sortie de l'étape précé-

dente et vérifie sur chaque point si un objet est présent. Pour cela, il utilise plusieurs “Anchors” de tailles différentes que l'on peut paramétrer si nécessaire. On peut voir plusieurs exemples d'anchors ci-dessous.

FIGURE 3.2 – Anchors

Chaque anchor est vérifiée pour renvoyer en sortie une liste de propositions d'objets sous la forme de plusieurs bounding box ayant le meilleur score.

FIGURE 3.3 – Bounding boxes

La troisième partie ou ‘Region of Interest Pooling’ (ROI), prend en entrée la feature map du deuxième backbone et les bounding box en sortie du RPN. Celles-ci sont alors extraites sur la feature map puis divisées en sous régions. Cela permet de réaliser un max pooling sur ces sous régions afin de leur donner une taille fixe.

La quatrième et dernière partie, ‘fully connected’, est composée de deux phases : l'une de classification permettant de connaître la classe des différents objets et l'autre de ‘bounding box régression’ qui permet d'affiner les bounding boxs.
Le schéma ci-dessous illustre la mise en place du modèle dans son intégralité. Pour des raisons de performance, les parties RPN et ROI sont réalisées en parallèle.

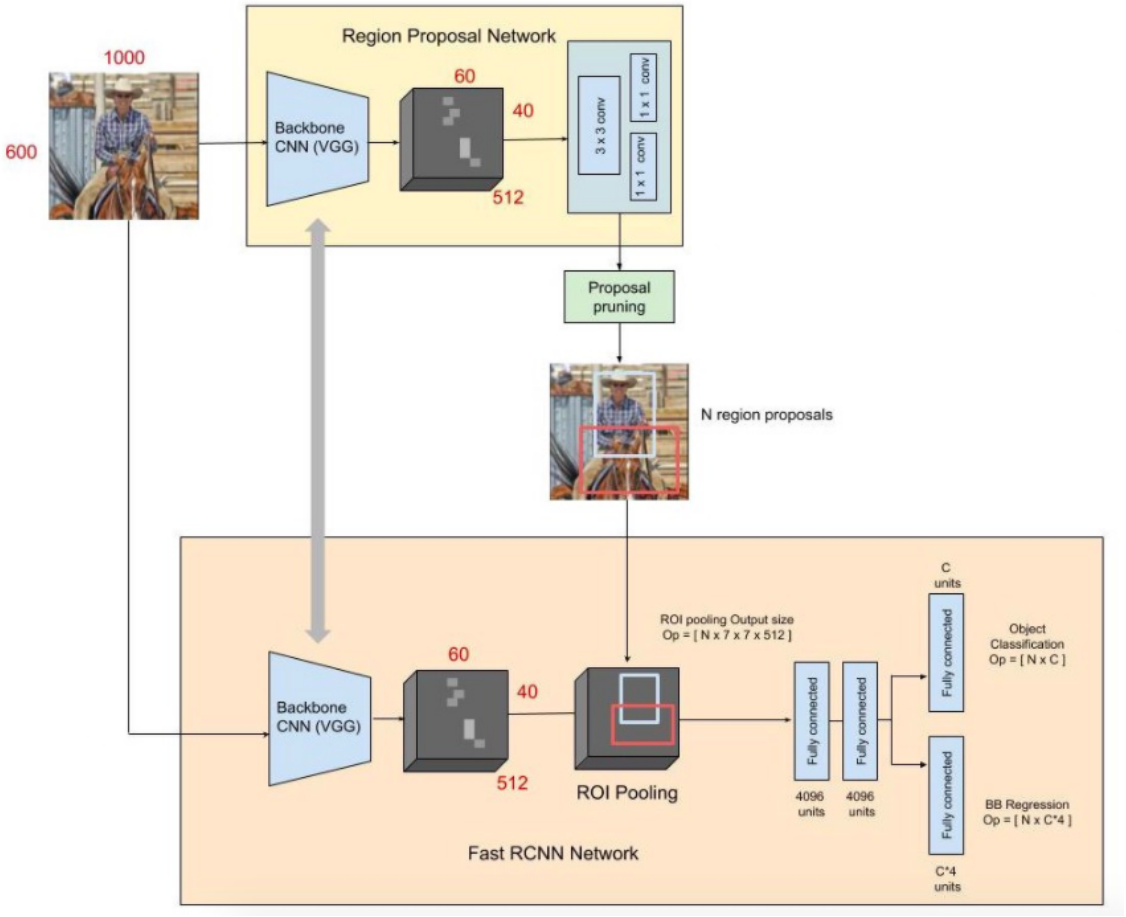

FIGURE 3.4 – Schéma du modèle de détection

3.2 Modèle de détection d'objets : prérequis

Pour ce modèle, nous avons bien sûr dû créer un nouveau jeu de données constitué d'images de plaques de hiéroglyphes, et non plus de hiéroglyphe unique, ainsi que d'annotations permettant de situer les hiéroglyphes sur ces plaques. Ayant trouvé de la documentation permettant de réaliser le modèle de détection grâce au format COCO, nous avons choisi ce format de données. Pour construire ce jeu de données, nous avons téléchargé une centaine d'images de plaques et les avons annotées grâce au logiciel LabelImg dans le format Pascal Voc. Nous avons utilisé ce logiciel car c'est une référence dans le domaine pour annoter les images. Pour finir, nous avons converti le jeu de données dans le format coco grâce à un script python.

Là encore, nous avons utilisé le 'transfer learning' pour améliorer les performances du modèle en adaptant l'étape backbone. Dans un premier temps, nous avons essayé de reprendre notre modèle de classification, mais VGG16 était trop lourd pour être utilisé dans ce cas-là. Nous avons donc choisi un autre modèle pré-entrainé (mobilnet_v2), moins performant sur le modèle de classification mais capable d'être entrainé sur nos images et intégré au modèle de détection.

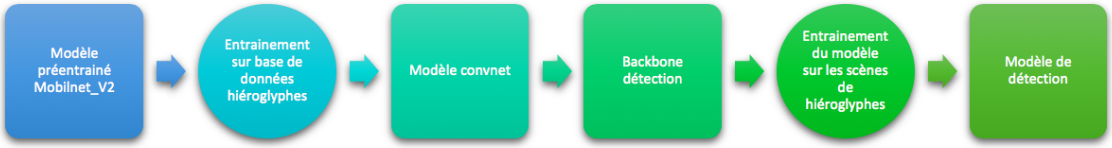

FIGURE 3.5 – Pipline du modèle de détection prévu

Toutefois, les résultats obtenus ainsi n'étaient pas satisfaisants : les données de la base d'entraînement étaient trop faibles pour obtenir de bonnes performances. En effet, nous avons pour le moment annoté seulement les hiéroglyphes reconnus également par notre modèle de classification, c'est-à-dire dix signes différents. Nous avons donc finalement utilisé mobilnet_v2 tel quel.

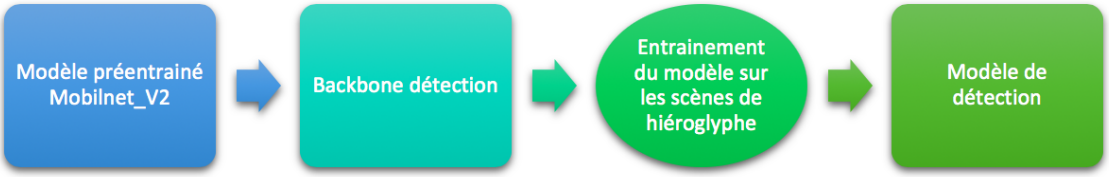

FIGURE 3.6 – Pipeline du modèle de détection réalisé

3.3 Modèle de détection d'objets : évaluation et perspectives

La principale difficulté rencontrée sur ce modèle concerne la taille des images qui sont extrêmement grandes, avec des hiéroglyphes souvent de mauvaises qualité. Les performances étaient alors très faibles. La solution que nous avons employée a été de découper les plaques afin de réduire la taille des images en entrée du modèle sans perdre en qualité de l'image. Cela nous a permis d'améliorer les résultats du modèle. La même technique est utilisée sur le site : lorsqu'une image est téléchargée, elle est automatiquement découpée avant l'analyse puis reformée une fois analysée.

Pour mesurer les performances du modèle, nous avons utilisé la métrique COCO-style mAP (mean Average Precision) qui mesure la précision du modèle quant à la classification et sa capacité à situer correctement les bounding box. Le résultat fourni par cette métrique se situe entre 0 et 1. Dans notre cas, il était de 0.24 ce qui reste faible. Toutefois, nous nous sommes rendus compte que le modèle parvient à localiser des hiéroglyphes et à les labelliser correctement. En réalité, le score du modèle est faible parce qu'il ne trouve pas tous les hiéroglyphes présents sur la plaque alors que les bounding box et le label associé sont bons. Le problème est donc probablement dû une fois encore à un manque de données. Le fait de devoir utiliser une backbone moins performante pour pouvoir entraîner le modèle contribue également à réduire les performances du modèle.

En perspective, il serait donc intéressant d'utiliser l'active learning sur ce modèle de détection de hiéroglyphes afin d'en augmenter les performances. L'expert pourrait alors donner son avis sur la détection proposée par le modèle : Est-elle correcte ? Suffisante ? Si oui, l'image pourrait être stockée dans la base de donnée. Sinon, l'expert aura la possibilité, sur l'onglet labellisation du site web, de sélectionner à

l'aide de sa souris les hiéroglyphes et de les labelliser un à un. Cette fonctionnalité permettrait d'agrandir notre jeu de donnée et ainsi d'améliorer les performances du modèle.

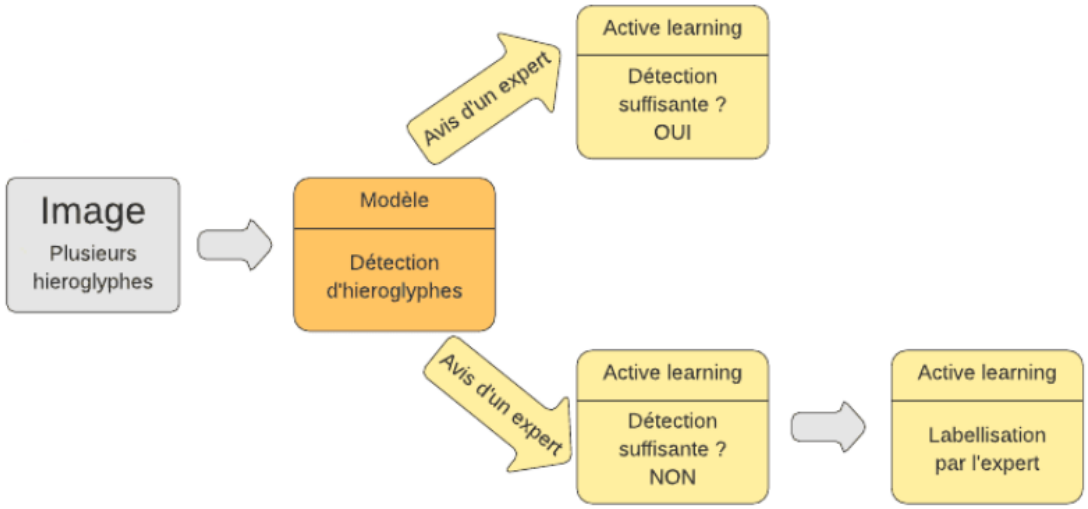

FIGURE 3.7 – Pipeline de l'active learning du modèle de détection

Pour conclure sur les différents modèles développés dans ce projet, voici la pipeline de ce qui a réellement été mis en place :

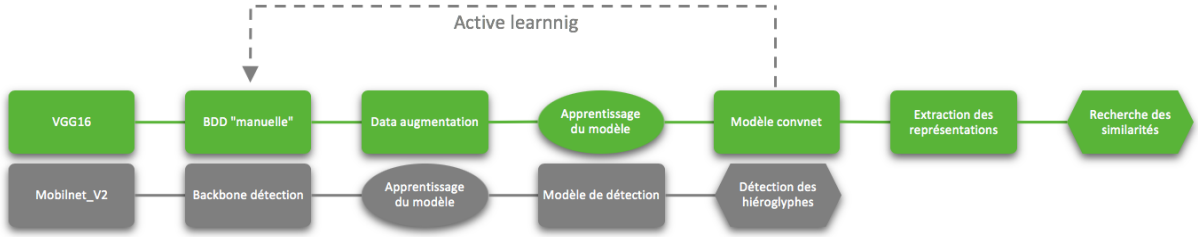

FIGURE 3.8 – Pipeline globale finale

Chapitre 4

Le site web

Le site web est l'interface permettant aux utilisateurs (experts) d'accéder aux différentes fonctionnalités que nous avons mises en place. Il se compose de sept onglets permettant de répondre à nos objectifs et à la demande du commanditaire Sebastien Biston-Moulin. Dans cette partie, nous présentons tout d'abord l'interface du côté utilisateur, puis sa mise en place technique et enfin les perspectives d'amélioration.

4.1 Site web : côté utilisateur

A l'ouverture du site, l'utilisateur doit s'identifier, ou s'inscrire s'il s'agit de sa première connexion, afin d'accéder aux différents onglets. Cette connexion lui permet de bénéficier d'un historique des différentes recherche effectuées. C'est aussi une manière de réserver l'accès aux spécialistes du domaine, permettant notamment aux utilisateurs d'enrichir la base de donnée avec un système de labellisation libre d'accès.

Une fois connecté, l'utilisateur est redirigé vers la page d'accueil présentant le site et ses fonctionnalités. Il est également possible d'en savoir plus sur le contexte de création du site et sur la source des données utilisées dans la page "A propos".

Dans les onglets "Analyse" et "Détection" se trouvent les outils créés pour répondre à la demande du commanditaire.

Dans le premier, l'utilisateur peut ainsi télécharger une photo d'un hiéroglyphe unique et il verra alors le résultat de sa recherche : la prédiction du code d'identification du signe et les cinq images les plus similaires de la base de données. Il s'agit bien sûr d'images contenant un signe ressemblant, souvent identique, mais présenté dans des contextes différents.

Dans le second, l'utilisateur peut soumettre une photo contenant plusieurs hiéroglyphes et il obtiendra pour chaque signe identifié un marqueur rectangulaire rouge associé à une prédiction du code d'identification.

L'onglet "labellisation" permet au spécialiste de participer au développement de la base de données (permettant ainsi l'amélioration des outils). Le principe est de proposer à l'utilisateur de labelliser les signes dont le résultat obtenu lors de l'analyse n'était pas fiable (cf. Active learning, Chapitre 2, section 3). L'objectif est à la fois de s'assurer d'intégrer des nouvelles images correctement classées mais aussi

de progressivement agrandir le nombre de signes connus par l'outil. Pour cela, l'utilisateur a simplement à renseigner le code associé à l'image qui lui est présentée lorsqu'il le connaît.

L'onglet "historique" permet grâce au système de connexion d'accéder à l'historique des recherches de l'utilisateur. Il y retrouvera donc les images qu'il a soumises à l'analyse ainsi que les résultats obtenus : le code d'identification associé au signe par le modèle. Cela lui permet de retrouver les signes déjà soumis au cours de précédentes visites sur le site web, ou de vérifier les résultats obtenus alors.

Enfin, l'onglet "bibliothèque" référence les différents hiéroglyphes reconnus sur le site web. Cette bibliothèque est donc vouée à s'agrandir avec le temps grâce à l'activité des utilisateurs.

La vidéo de démonstration du site web jointe à ce document vous permettra de visualiser le résultat du site web et d'identifier toutes les fonctionnalités présentées ci-dessus.

4.2 Site web : Outils utilisés

Pour proposer des onglets fonctionnels, notre site web nécessitait d'être connecté aux modèles de classification, de détection et de recherche des similarité. Ces derniers ayant été réalisés en python, il nous a donc fallu dans un premier temps trouver un moyen de lier notre interface web à des codes python. De plus, nous devions utiliser un outil professionnel permettant potentiellement de mettre le site en production et de le publier. Ces différentes contraintes nous ont amenés à choisir l'outil Flask, un Framework disponible sur python permettant de faire du développement web. Il permet de lier les pages html aux modèles de façon simple grâce à un backend en python, avec la possibilité de réaliser le design du site en CSS.

Dans un second temps, nous avons dû réfléchir à un moyen de gérer les sessions des utilisateurs pour plusieurs raisons. En effet, seul un expert peut participer à l'active learning grâce à sa connaissance des hiéroglyphes, sans quoi la base de données pourrait rapidement être biaisée. De plus, cela donne la possibilité de proposer un historique des analyses effectuées sur le site. Nous avons donc mis en place un système de connexion nécessitant l'utilisation d'un identifiant (adresse mail) et d'un mot de passe pour avoir accès au site.
L'historique étant propre à chaque utilisateur, nous devions associer un utilisateur à son activité (i.e. aux images téléchargées) sur le site. Pour cela, nous avons créé une base de données dédiée aux utilisateurs : chacun d'eux est associé à un mot de passe, un email et une liste d'images analysées. Nous nous sommes ensuite naturellement tournés vers les extensions de Flask permettant de créer un système de sessions : flask_login et flask_alchemy. Flask_login permet de gérer des sessions des utilisateurs tandis que Flask_alchemy permet de gérer la base de donnée en simplifiant l'utilisation de SQLalchimy avec Flask.

L'historique fonctionne de la façon suivante : lorsque l'utilisateur lance une analyse sur un hiéroglyphe dans la partie analyse, notre modèle retourne les images similaires mais également la prédiction de la classe du hiéroglyphe. Ces données sont enregistrées puis affichées sous forme de galerie dans l'onglet historique. L'onglet active learning quant à lui ne présente les images à labelliser qu'une à une, en commençant par celle qui a l'entropie de shannon la plus faible. L'utilisateur peut alors choisir, dans une liste déroulante, le code correspondant au hiéroglyphe présent sur l'image. Le choix de la liste déroulante

a été fait de manière à éviter les fautes de frappe qui auraient une incidence sur le classement des nouvelles images au moment de les enregistrer dans la base de données. Toutefois, il est également possible pour l'utilisateur d'entrer du texte pour pallier le cas où le hiéroglyphe soit nouveau sur le site et donc pas encore répertorié dans la base de données associée.

4.3 Site web : Limites et perspectives

Si le site web s'adresse aux utilisateurs experts en égyptologie, pour l'instant, n'importe quel utilisateur peut s'inscrire via l'utilisation d'une adresse email. Nous avons donc pensé mettre en place un système de code généré par l'administrateur qui permettrait l'inscription à la demande. Seules les personnes connues et validées par l'administrateur pourraient alors accéder au formulaire d'inscription. Il pourrait aussi être intéressant de rendre le site accessible au plus grand nombre pour aider notamment à l'enrichissement de la base de données. Toutefois, l'historique ainsi que la restriction de l'accès à l'onglet labellisation restent nécessaires. Nous pourrions donc faire la distinction entre les utilisateurs "experts" et les "non experts" au moment de l'inscription, par le renseignement de leur profession ou l'usage d'un code demandé à l'administrateur du site. Ainsi, tous pourraient avoir un historique de recherche mais seuls les experts auraient accès à la labellisation des hiéroglyphes non identifiés.

De plus, actuellement, l'onglet historique ne renvoie que les images téléchargées dans l'onglet analyse et non celles sur lesquelles la détection de hiéroglyphes est effectuée. Il s'agit d'une perspective directe et importante à mettre en place sur le site pour rendre l'historique plus complet et plus précis.

Enfin, une application mobile pourrait être développée afin de pouvoir "scanner" via la caméra de son smartphone les plaques de hiéroglyphes et obtenir le résultat des suggestions en temps réel. Cela rendrait les fonctionnalités du site web peut-être plus accessibles à tous et permettrait à l'expert de scanner directement le signe ou la scène qui l'intéresse lors d'une visite au temple de Karnak par exemple.

Chapitre 5

La gestion de projet

Le mode de gestion est primordial dans la réussite d'un projet. De nombreux chercheurs ont travaillé sur le sujet pour savoir quels facteurs sont les plus importants à prendre en compte dans le mode de gestion de projet afin de rendre sa mise en place la plus efficace possible. Certaines méthodes font aujourd'hui partie de notre quotidiens, on les appelle "Agile" ou encore "Scrum". Chaque entreprise choisit et adapte à sa façon ces modes de gestion. Pour notre projet de TER, nous ne nous sommes pas cantonnés à utiliser une méthode en particulier mais avons décidé de nous inspirer de celles-ci. Nous allons donc voir comment nous fonctionnons aujourd'hui dans notre travail de groupe mais également comment nous avons su apprendre de nos erreurs et évoluer au cours du projet.

5.1 Premiers pas dans la gestion de projet informatique

Avant de présenter notre mode de fonctionnement, il semble important de présenter les membres du groupe et notamment leurs compétences spécifiques. En effet, les membres de l'équipe sont l'élément essentiel à connaître pour une bonne répartition des tâches. Nous ne sommes pas tous issus des mêmes formations, par exemple. Deux personnes sont issues de la licence MIASHS de l'Université Paul Valéry, Sarah et Gaëlle ; deux autres d'une licence professionnelle "Chargé d'Études Statistiques", Fleur et Priscille ; et enfin Théo sort d'une licence professionnelle DevOps. Nous avons là plusieurs compétences différentes, certaines plutôt orientées statistiques, d'autre développement/programmation et enfin d'autres compétence relatives à la conduite d'étude et aux domaines d'application.

Etant donnée l'amélioration de notre mode de gestion de projet au cours de l'année, une analyse critique de la gestion de projet du premier semestre semble pertinente. Ainsi, au début de notre projet, notre organisation était la suivante.
Les réunions entre membre du groupe étaient fixées pour la semaine même lors de notre retour à l'université, ce qui n'était alors pas optimum en ce qui concerne l'organisation de tous les membres de l'équipe pour être présent. L'ordre du jour n'était alors pas toujours clairement défini, ce qui pouvait parfois mener à une perte de temps pour se centrer sur le sujet. De même, les réunions avec les commanditaire et tuteur n'étaient souvent pas assez anticipées.
En ce qui concerne la communication au sein du groupe, celle-ci était plutôt claire mais semblait parfois se compliquer un peu lors de discussions plus techniques. Cela était alors dû aux différences de connaissances des membres du groupe, notamment liées au vocabulaire utilisé ainsi qu'aux connaissances de base de chacun en deep learning. Il a donc fallu un temps d'adaptation et de compréhension plus important

pour certains membres du groupe.
La répartition des tâches était quant à elle effectuée uniquement à l'oral. Cela pouvait alors poser problème en cas de quiproquo et d'incompréhension de certains membres de l'équipe, ou provoquer une perte de temps lorsque plusieurs personnes avaient travaillé en parallèle sur le même point. De plus, nous n'avions pas de façon concrète de visualiser l'avancée de notre travail. Afin de remédier à cela, nous avions mis en place un fichier partagé 'Google Doc' sur lequel nous notions les sujets abordés lors des réunions et les tâches à effectuer. Cependant, ces missions n'étaient pas clairement attribuées à une personne en particulier. De plus, cet outil n'était pas vraiment pratique et tout simplement pas adapté au suivi du projet et des missions en cours. Afin de travailler de façon collaborative, nous avons créé un GitHub. Celui-ci nous permettait alors de partager nos fichiers afin que tout le monde y ait accès. Cependant, nous n'avons pas utilisé ce logiciel de gestion de version de façon optimale, notamment en ce qui concerne la gestion des codes python. En effet, l'organisation des dossiers n'était pas très claire et les différents "morceaux" de code ne pouvaient pas être lancés tous en même temps. Il fallait donc modifier des fichiers pour tester l'ensemble, ce qui, il est vrai, n'était pas des plus pratique. Enfin, nous nous sommes également aperçu que des rendez-vous plus fréquents avec notre tuteur pourrait nous permettre d'évoluer plus efficacement et de nous voir accorder quelques précieux conseils.

Cette première analyse de gestion de projet peut sembler plutôt négative, cependant, nous avons fait notre possible afin d'évoluer et de gérer notre projet au mieux. Grâce à notre expérience, aux différents conseils et aux interventions de spécialistes en gestion de projet, nous avons pu modifier notre façon de procéder pour la seconde partie du projet, ce que nous allons voir maintenant.

5.2 Vers une amélioration du mode de gestion de projet

Malgré une gestion de projet très peu élaborée lors du premier semestre, l'avancée du projet a tout de même été efficace et les tâches prévues ont été réalisées dans les temps. Afin d'être malgré tout plus productifs et avoir une meilleure communication et cohésion de groupe, nous nous sommes ensuite organisés de la manière suivante en termes de gestion de projet.

Dans un premier temps, il semble évident que tous les membres du groupe ont été, durant cette période, plus à l'aise en ce qui concerne le sujet et les technologies à utiliser pour l'améliorer. En effet, la formation ayant commencé, nous n'avons plus rencontré de problème de communication au niveau technique, ce qui a été décisif dans le bon avancement des différentes tâches. Cependant, afin de mieux coordonner les différentes missions et tâches à effectuer, nous avons désigné un chef de groupe. En effet, un projet "est opéré par une équipe projet, ce qui induit méthode, organisation et définition d'un chef de projet". C'est ainsi que Théo a pu prendre un rôle de supervision du groupe, définissant les tâches de chacun en prenant en compte les compétences des membres du groupe. Le choix du chef de projet s'est fait en fonction des compétences : nous avons choisi la personne la plus à l'aise avec le sujet et donc la plus à même d'évaluer les délais incombant au périmètre du projet. En effet, dès le premier semestre, il s'était déjà naturellement positionné en tant que tel, mais "l'officialisation" de ce rôle lui a permis de prendre davantage position et d'être pris plus au sérieux encore. Ouvert aux échanges, il a pu être à l'écoute de chacun des membres du groupe, afin de les aider dans leur avancement si besoin. Bien qu'un chef de groupe ait été désigné pour le management des tâches, les différents choix concernant le projet ont été effectués entre tous les membres du groupe. Théo a alors eu un rôle de médiateur dans les échanges

et avait pour mission, lorsque c'était nécessaire, de trancher lors d'avis divergents ou de malentendus. Lors des réunions d'équipe, le chef de projet prenait la parole en premier afin de rappeler l'ordre du jour. Cette manière de procéder nous a permis d'être plus efficace lors des réunions et d'écourter leur durée ou d'en profiter pour débattre plus longtemps sur des sujet importants tels que le choix des outils et les tâches à prioriser.

Plus précisément, en ce qui concerne les réunions de groupe, celles-ci ont été décidées plus à l'avance que lors du premier semestre afin de faire en sorte que tous les membres du groupe puissent être présents et préparés à chaque réunion. Cela est important puisque les réunions permettent à chacun de faire part de leur avancement sur leur(s) tâche(s) mais aussi de définir les projets pour la suite, répondre aux questions. Comme vu précédemment, nous avons essayé de définir l'ordre du jour pour chaque réunion ainsi qu'une durée maximum d'une heure afin de conserver l'attention de tous les membres. Nous n'avons pas formellement mis en place de compte rendu de réunion, comme lors du premier semestre, ce qui pourrait être encore un point d'amélioration. Chaque membre du groupe prenait alors des notes individuellement. Nous avons essayé de tenir une régularité dans les réunions effectuées. Ainsi, la charte des réunions s'articulait de la façon suivante : une réunion était programmée entre les membres du groupe lors du début des périodes à l'université. Elle avait pour but de faire un compte rendu du travail à effectuer sur la période et des éventuels avancements réalisés par chacun pendant la période en entreprise. Une réunion à la fin de chaque période à l'université était également mise en place, plus courte, consistant alors à faire le point sur ce qui avait été réalisé lors de la période et du travail à fournir lors de la prochaine période. C'est lors de cette réunion que nous avons donc rédigé les différents comptes rendus d'avancement à destination du tuteur et du commanditaire. Celle-ci n'était pas vouée à régler des problèmes majeurs. Pour cela, une réunion spéciale était organisée au besoin. En ce qui concerne les réunions avec le tuteur universitaire, nous avons essayé de prendre un rendez-vous par période à l'université (d'une période pour la suivante) afin qu'il puisse avoir une meilleure vision de notre avancement et nous conseiller. Des réunions régulières avec le commanditaires n'ont pas été réalisées. En effet, nous avons effectué une réunion d'avancement afin de présenter le devenir du projet et s'assurer de notre bonne direction, cependant, il ne s'est pas avéré nécessaire de prévoir d'autres rendez-vous avec le commanditaire, celui-ci nous ayant rassuré sur notre avancée. Il s'agissait là d'éviter toute perte de temps pour le commanditaire et nous-mêmes. Lors de ces réunions (tuteur ou commanditaire), contrairement aux réunions d'équipe, une personne prenait des notes et les partageait à la fin de la réunion en guise de compte rendu.

Un projet consiste en un "ensemble d'activités coordonnées et maîtrisées ayant un début et une fin". Ces différentes activités et missions doivent donc être répertoriées clairement et assignées à des membres du groupe afin d'être en capacité de savoir quand celles-ci sont commencées et terminées mais également quelles sont celles qu'il reste à effectuer. Afin de suivre cela, nous avons mis en place l'utilisation d'un outil que nous n'avions précédemment pas utilisé sur GitHub : le mode projet. Cet outil permet une répartition plus claire des tâches à effectuer, pouvant être décrites dans un onglet spécifiquement dédié. Cela donne accès à un aperçu rapide pour chaque grande mission de ce qui a été effectué ou reste à faire. Au sein de chaque grande rubrique peuvent être distinguées les tâches "en cours", "à faire" et "terminées". L'utilisation de la fonctionnalité "issue" permet d'alerter les membres de l'équipe en cas de difficultés rencontrées sur la mise en place d'une tâche. Il est par ailleurs possible d'affecter une personne à la résolution du problème rencontré. Ainsi, un suivi rigoureux des tâches est possible. Globalement, les différentes tâches ont été répartie de la façon suivante entre les différents membres du groupe :

- Théo, Chef de projet, s'est concentré sur le modèle de détection et a su apporter de l'aide aux autres membres du groupe lorsqu'il y avait un blocage technique.
- Fleur a travaillé sur l'amélioration des techniques de data augmentation ainsi que sur le site web (mise en place de Flask, gestion des sessions).
- Priscille a réalisé le web scraping, cherché à améliorer le modèle de recherche de similarités et a également travaillé sur la mise en forme du site web (maquette, mise en forme du site, front/CSS).
- Gaëlle a travaillé sur le contenu du site web.
- Sarah a réalisé la partie active learning, permettant la labellisation automatique des images, et l'a liée au site web via l'onglet Labellisation.

Enfin, dès le mois de décembre, nous avons mis en place l'outil Slack afin de communiquer sur le projet. Cela nous a permis de disposer de conversations par thématiques et ainsi de ne pas perdre les messages importants. Par la suite, cet outil nous a été très utile en ces temps de confinement. En effet, nous avons pu lier l'application Slack au projet GitHub. Grâce à cela, nous pouvions être tenus informés des commits effectués par chacun. De plus, sur GitHub, nous avons réorganisé le code python de façon à ce que l'on puisse lancer tout le code ou non afin de faire des tests et ainsi d'utiliser GitHub de la bonne manière.

En conclusion, un projet c'est : "estimer, planifier, organiser, piloter et suivre l'avancement". Il semblerait que nous ayons, malgré quelques difficultés au démarrage, réussi à mener notre projet à bien. Nous avons en effet été en capacité de réaliser les différentes fonctionnalités du site promises au commanditaire. Nous avons donc rempli une grande partie du cahier des charges mais pouvons accuser de quelques améliorations restant à mettre en place. Cependant, dès le début du projet et la définition des objectifs, nous avons su mettre en garde le commanditaire sur la difficultés des tâches à réaliser et la possibilité de ne pas être en capacité de toutes les mettre en place.

Conclusion

Effectué dans le cadre du Master 1 MIASHS, ce travail s'inscrit dans le projet Karnak dirigé par Sebastien Biston-Moulin. L'objectif final était de fournir une interface offrant deux fonctionnalités : une première capable d'identifier un hiéroglyphe et de renvoyer des images de hiéroglyphe similaire et une seconde permettant de détecter et reconnaître un hiéroglyphe parmi un ensemble de signes.

La mise en place de l'interface, ici un site web, et des différentes fonctionnalités a exigé l'utilisation de modèles complexes et d'outils nouveaux pour nous tels que le modèle de détection d'objets Faster Rcnn ou le framework Flask par exemple.

Il existe bien sûr de nombreuses perspectives à ce travail, tant au niveau de l'enrichissement des données afin d'améliorer les performances des modèles, que de la mise en place d'autres fonctionnalités telles que l'active learning sur le modèle de détection ou encore de l'amélioration du site web avec par exemple un système d'inscription différenciant expert et non expert.

Toutefois, si les résultats pourraient encore être améliorés, nous avons pu concrétiser les objectifs fixés et ainsi répondre à la demande initiale tout en expérimentant diverses méthodes de gestion de projet dans le but d'un travail toujours plus efficient et professionnel.

Bibliographie

[1] Ananth, S. (2019). *Faster R-CNN for object detection*. Disponible sur : https ://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474c5b857b46

[2] Auclair, A., Cohen, L.D., Vincent, N. (2009). *Hachage de Descripteurs Locaux pour la Recherche d'Images Similaires*. Congres des jeunes chercheurs en vision par ordinateur (ORASIS'09), Tregastel, France.

[3] Bort, J. Emvista. (2020). *Référence du cours*. Gestion de projets. Recueil inédit, Université Paul Valery.

[4] CNAM RCP216 (2019). *Cours - Recherche par similarite. Application aux systemes de recommandation*. Disponible sur : http ://cedric.cnam.fr/vertigo/Cours/RCP216/coursSimilariteRecommandation.html

[5] Delbart, D. Sopra Steria. (2020). *Référence du cours*. Gestion de projets. Recueil inédit, Université Paul Valery.

[6] Gandhi, A. (2019). *How to use Deep Learning when you have Limited Data - Part 2*. Disponible sur : https ://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/

[7] Hari, S.S. (2018). *Locality Sensitive Hashing*. Disponible sur : https ://santhoshhari.github.io/Locality-Sensitive-Hashing/

[8] Herbert, A. (2019). *How To Add Authentication to Your App with Flask-Login*. Disponible sur : https ://www.digitalocean.com/community/tutorials/how-to-add-authentication-to-your-app-with-flask-login

[9] Hui, J. (2018). *mAP (mean Average Precision) for Object Detection*. Disponible sur : https ://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

[10] Khandelwal, R. (2019). *COCO and Pascal VOC data format for Object detection*. Disponible sur : https ://towardsdatascience.com/coco-data-format-for-object-detection-a4c5eaf518c5

[11] Luthon, F., Navarro, X., Liévin, M. (2010). *Seuillage Entropique En Traitement d'images*. 18e Colloque sur le Traitement du Signal et des Images (GRETSI'01), Toulouse, France.

[12] Nougaillac, M. Rectorat de Montpellier. (2020). *Référence du cours*. Gestion de projets. Recueil inédit, Université Paul Valery.

[13] Pytorch tutorials (2017). *Torchvision object detection finetunning tutorial*. Disponible sur : https ://pytorch.org/tutorials/intermediate/torchvision_tutorial.html

[14] Sarthak, J. (2017). *How to use Deep Learning when you have Limited Data*. Disponible sur : https ://medium.com/nanonets/nanonets-how-to-use-deep-learning-when-you-have-limited-data-f68c0b512cab

Table des figures