Rapport de Projet

Partie 3

Jacob Maurice (300187393)

Département de Science Informatique
Université d'Ottawa

Instructeur du Cours
Robert Laganière

Date de Soumission
Le Mercredi, 10 avril 2024

Le rapport présent a pour objectif de détailler la formulation d'un algorithme visant à associer plusieurs images à une personne à l'aide d'histogrammes.

Détails d'exécution

Partie 3

Conforme au script Python intitulé "part3.py" fourni dans le dossier de soumission, l'algorithme repose sur l'utilisation de plusieurs bibliothèques, notamment OpenCV, NumPy et Pillow (ainsi que les modules os et shutil). Il est donc impératif de vous assurer d'avoir installé les dépendances nécessaires si vous souhaitez exécuter ce fichier.

Voici les commandes pip pour l'installation des bibliothèques requises :

Installation d'OpenCV
pip install opencv-python

Installation de NumPy
pip install numpy

Installation de Pillow
pip install pillow

Il est également important de noter que le script doit être exécuté sans spécifier d'arguments dans la ligne de commande, et il générera 5 fichiers contenant les 100 meilleures images pour chaque individu spécifié dans les images de test.

Veuillez prendre note qu'il est impératif d'exécuter le fichier "main.py" avant "part3.py" si le fichier de sortie nommé "output" n'a pas été encore rempli avec les données provenant de l'algorithme attribuant un masque rouge aux individus présents dans chaque image d'entrée. En outre, assurez-vous également que le fichier "bounding_boxes.txt" existe et qu'il contient

les coordonnées de la boîte englobante de chaque individu dans les images d'entrée, selon le même format que le fichier nommé “labels.txt” dans la partie 1 du projet.

Formulation de l’algorithme

L'algorithme implémenté se divise en deux fichiers Python : "part3.py", dont le code demeure largement inchangé depuis la partie 1 du projet, et "main.py", qui doit être exécuté en premier lieu pour générer un fichier "output" contenant toutes les images des fichiers d'entrée "cam0" et "cam1" après l'application d'un masque de détection sur les personnes présentes. Il est à noter que l'algorithme fonctionne de manière similaire à la partie 2 du projet, mais des modifications ont été apportées au fichier "tools.py" dans le répertoire "utils". Désormais, celui-ci génère un masque prenant la forme de la boîte englobante entourant chaque individu dans l'image.

Voiçi le résultat visuellement:

Figure 1 - Masques ayant la forme d’une boîte englobante

Bien que cette approche ne soit pas aussi efficace, le traitement des listes numpy pour conserver une collection des pixels de chaque individu masqué dans chaque image a entraîné divers problèmes que je n'ai pas eu le temps de corriger en raison de contraintes temporelles. Par conséquent, j'ai également modifié l'algorithme pour qu'il produise un fichier texte nommé "bounding_boxes.txt", contenant une liste des paramètres nécessaires pour définir la boîte englobante de chaque individu dans chaque photo, afin que son format soit identique à celui du fichier "labels.txt" de la partie 1 du projet.

Le fichier "main.py" reste globalement inchangé depuis la partie 2 du projet.

Conformément à ce qui a été présenté dans la première phase du projet, j'ai recouru au code Python "part3.py", auquel j'ai apporté quelques modifications le distinguant de "part1.py". Le code fait l'usage désormais du fichier "bounding_boxes.txt" au lieu de "labels.txt" afin d'initialiser la boîte englobante de chaque individu. De plus, les images de test nommées "person_x" sont dorénavant employées en lieu et place des deux images de test présentées dans la première phase. Finalement, les fichiers contenant la liste des images non altérées sont désormais désignés sous les noms "cam0" et "cam1".

Toutefois, la fonctionnalité du code demeure la même et l'algorithme est présenté de la même façon qu'en première phase, décrit ci-dessous:

Le code Python "part3.py" inclus dans le dossier de soumission propose l'implémentation d'une pipeline de traitement d'images. Celle-ci est élaborée spécifiquement afin de conduire l'analyse des individus apparaissant dans des boîtes englobantes repérées dans l'avant-plan d'une image de référence. Son objectif principal est d'identifier les 100 boîtes englobantes les plus similaires parmi l'ensemble des images constituant une séquence vidéo donnée, de façon à ce qu'elle corresponde à ces personnes ayant été identifiées.

Dans un premier temps, la pipeline initialise un dictionnaire (nommé "image_dict") qui associe les noms des images de la séquence vidéo (utilisés comme clés) à leurs représentations (en objet PIL.Image). Ce dictionnaire est peuplé en parcourant les images dans des dossiers (nommés "cam 0" et "cam 1") spécifiés à l'aide de la fonction

"create_image_dict". Cette étape préliminaire établit les fondations pour les opérations de traitement d'images ultérieures.

Suivant la création du dictionnaire d'images, la pipeline procède à l'extraction des boîtes englobantes à partir de l'ensemble des images de la séquence vidéo. Cette opération consiste à lire les coordonnées des boîtes englobantes depuis un fichier texte (nommé "bounding_boxes.txt") et à utiliser des fonctions telles que "extract_bounding_box" et "extract_upper_half_bounding_box" afin de découper et stocker les régions correspondantes des images dans deux listes distinctes, en tenant compte du fait que la première liste comprend toutes les boîtes englobantes de la séquence tandis que la deuxième liste ne contient que les moitiés supérieures des boîtes englobantes. Veuillez noter que les boîtes englobantes dont la surface est inférieure à 2500 sont ignorées, car elles correspondent aux individus identifiés en arrière-plan.

Une fois les images des boîtes englobantes obtenues, la pipeline calcule des histogrammes basés sur l'espace de couleur HSB pour chaque image découpée à l'aide de la fonction "calc_histogram". Ces histogrammes capturent la distribution des pixels dans les images en fonction de leur teinte, leur saturation et leur brillance sur une échelle de 256 valeurs, ce qui est crucial pour la comparaison des images en fonction de leur contenu visuel.

Pour faciliter le processus de comparaison qui suivra, la pipeline sélectionne des images de test et extrait leurs boîtes englobantes. Celles-ci servent de référence pour comparer les individus spécifiés avec ceux présents au sein des autres images de la séquence. Ainsi, des histogrammes découpés (en fonction des dimensions de leur boîte englobante associée) sont générés pour ces images, facilitant ainsi la comparaison avec le reste de l'ensemble de données.

Une fois les histogrammes calculés, la pipeline les compare pour identifier les similitudes entre les individus dans différentes images. La fonction "compare_histogram_pairs" calcule les scores de similarité entre les histogrammes en utilisant la méthode d'intersection. Cependant, pour chaque individu de l'image test, il est nécessaire de comparer l'histogramme de sa boîte englobante ainsi que celui de sa moitié supérieure avec ceux de toutes les autres images de la séquence. Ainsi, une comparaison entre deux individus nécessite 4 comparaisons distinctes (définies dans le script comme "comp1", "comp2", "comp3" et

"comp4"). Cette opération est répétée pour tous les individus de la séquence pour chaque personne testée, et la pipeline sélectionne ensuite les 100 meilleures images ayant les scores de similarité les plus élevés. Cela est réalisé grâce à des fonctions telles que "get_top_100_images", qui trient les scores de similarité et récupèrent les images correspondantes. Enfin, les images sélectionnées sont enregistrées dans un dossier de destination à l'aide de la fonction "save_top_100_images", en vue d'une analyse ou d'une visualisation ultérieure.

Résultats

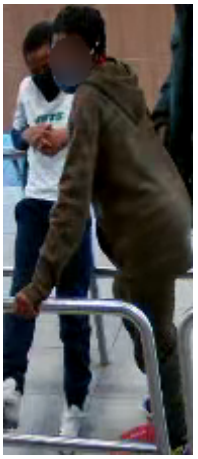
Étant donné que les résultats obtenus lors de la première phase se sont révélés assez probants, j'ai pris la décision de continuer à utiliser les histogrammes dans l'espace HSB, en incorporant la teinte, la saturation et la luminosité.

1. Résultats - Teinte, Saturation, et Brillance (Espace HSB)

	Images Test				
Personne à identifier	**Person_1**	**Person_2**	**Person_3**	**Person_4**	**Person_5**
Pourcentage de succès	74%	95%	99%	43%	9%

Il est évident que les résultats varient considérablement en fonction de la personne observée. Bien que les performances soient satisfaisantes pour les individus 2 et 3, il y a des possibilités d'amélioration pour les autres. La principale source de problème réside dans l'utilisation des boîtes englobantes plutôt que des contours précis des individus. Plusieurs défauts sont apparus. Dans le cas du premier individu, des résultats erronés sont dus au chevauchement partiel avec d'autres clients, faussant ainsi les résultats.

Figure 2 - Chevauchement de deux individus dans une seule boîte englobante



De plus, pour les individus 4 et 5, dont la silhouette est petite par rapport à la boîte englobante, l'environnement biaise les résultats de manière significative. Cela est dû au fait que ces individus apparaissent dans d'autres séquences de la scène, mais leur silhouette occupe une proportion beaucoup plus grande dans leur boîte englobante respective, ce qui génère des histogrammes totalement différents et ne permet pas leur reconnaissance en tant que la même personne.

En conclusion, une amélioration simple consisterait à exporter les pixels de chaque personne masquée dans les images au format numpy, puis à les utiliser dans l'algorithme de correspondance. Cependant, bien que cette approche ait été envisagée dès le départ, des difficultés ont été rencontrées lors de sa mise en œuvre, ce qui a conduit à opter pour les boîtes englobantes par manque de temps.