

Rapport Projet **Traitement d'image et** **vidéo**

Réalisé par :

HARROUZ WAIL

BOUHANIK YOUNES

Les descripteurs utilisé dans le projet :

- SIFT
- COMPACITÉ
- ASPECT RATIO
- Disparcité

Les algorithms utilisé dans le projet :

- PCA
- SVM et MLP pour la classification










les bibliothèques utilisées:

- sklearn
- pickle
- functools

NOTE : l'apprentissage était appliquer sur les données binaire pour pouvoire appliquer après sur un vidéo en utilisant un modèle de soustraction d'arrière plan

Tâche 1 : Préparation des données

Pour faciliter la tâche de “features extractions “ pour la préparation de vecteur d’input dans le modèle d’apprentissage

 bike	23/01/2020 22:45	Dossier de fichiers
 boats	23/01/2020 22:45	Dossier de fichiers
 canoe	23/01/2020 22:45	Dossier de fichiers
 car	23/01/2020 22:45	Dossier de fichiers
 human	23/01/2020 22:45	Dossier de fichiers
 noise	23/01/2020 22:45	Dossier de fichiers
 pickup	23/01/2020 22:45	Dossier de fichiers
 truck	23/01/2020 22:45	Dossier de fichiers
 van	23/01/2020 22:45	Dossier de fichiers

du sort que chaque fichier contient des images associé à une classe de output

Tâche 2 : SIFT descripteurs

Pour cette tâche on a utilisé l’extracteur de opencv
(on sélection N premier éléments par rapport à leurs distinctivité dans le local
)

Tâche 3 : Préparation du modèle PCA

Supposant que pour chaque objet (cropped image) on prend 20 points sift(les plus importants en terme de distinctivité)

ça implique que au moins on aura com me vecteur d’input de taille
 $50 \times 128 = 6400$

Donc nous avons appliqué la PCA pour réduire les dimensions des descripteurs , le nombre de composants choisis etais 8 donc le vecteur d’input sift (pour 50 points)

$$50 \times 8 = 400$$

donc nous avons réduit la taille de vecteur d’input de 6400 à 400

Finallement :

Sauvegarder le modèle de PCA pour transformer les inputs dans le nouvelle systems des coordonnées (réduit) et même pour les nouveau points à tester plus tard

Exemple :

un descripteur avant le PCA :

```
[ 23.  1.  0.  0.  0.  0.  0.  2. 178. 12.  0.  0.  0.  0.
  0. 27. 177.  5.  0.  0.  0.  0.  0. 12.  1.  0.  0.  0.
  1.  1.  0.  0. 43.  0.  0.  0.  0.  0.  0.  5. 178.  3.
  0.  0.  0.  0.  0. 64. 177.  1.  0.  0.  0.  0.  0. 25.
  1.  0.  0.  0.  0.  0.  0.  0. 45.  5.  0.  0.  0.  0.
  0.  0. 178. 50.  0.  0.  0.  0.  0.  6. 173. 20.  0.  0.
  0.  0.  0.  2.  1.  0.  0.  0.  0.  0.  0.  0. 18.  2.
  0.  0.  0.  0. 16. 14. 178. 35.  0.  0.  0.  0. 16. 56.
156. 24.  0.  0.  0.  0.  0.  9.  1.  0.  0.  0.  0.  0.
  0.  0.]
```

Après l'application du PCA :

```
[151.87988797 -39.30299904 173.25496368 -60.75708326 196.33372221
-38.86899379 48.6523593 -9.3509859 ]
```

Tâche 4 : Préparation du vecteur des inputs

**Après la construction du modèle PCA et l'extracteur de sift
pour chaque input (image binaire) on extrait les 50 points SIFT (leurs
descripteurs) et pour chaque descripteur on applique la PCA et donc nous
allons avoir 50 descripteurs de taille = 8 pour chaque image + le descripteur
de compacité et aspect ratio et la disparité**

Tâche 5 : Apprentissage

Dans cette partie on a fait deux classifieurs différents

un SVM et un MLP (multi-layer-perc)

Les meilleurs résultats obtenu avec 70% données pour l'apprentissage et 30 % pour les tests :

SVM :

- Prédications : 82.5%
- Loss : 0.21

MLP : (avec hidden layers (100,2) et solver = "lbfgs")

- Prédications : 91%
- Loss 0.013

à la fin on sauvegarde le classifieurs avec les meilleurs caractéristiques pour l'utiliser dans la tâche suivante

Tâche 6 : Partie test

Le fichier contient 3 parties de code :

- Partie pour tester l'apprentissage
- Partie pour les tests unitaire
- Partire pour tester sur une séquence d'image

Test unitaire :

Comme l'algorithme d'apprentissage était basé sur des images binaires donc faut mieux tester des images pour avoir les resultats

NOTE : (cette partie était faite juste pour vous montrer les résultats sur des images unitaires)



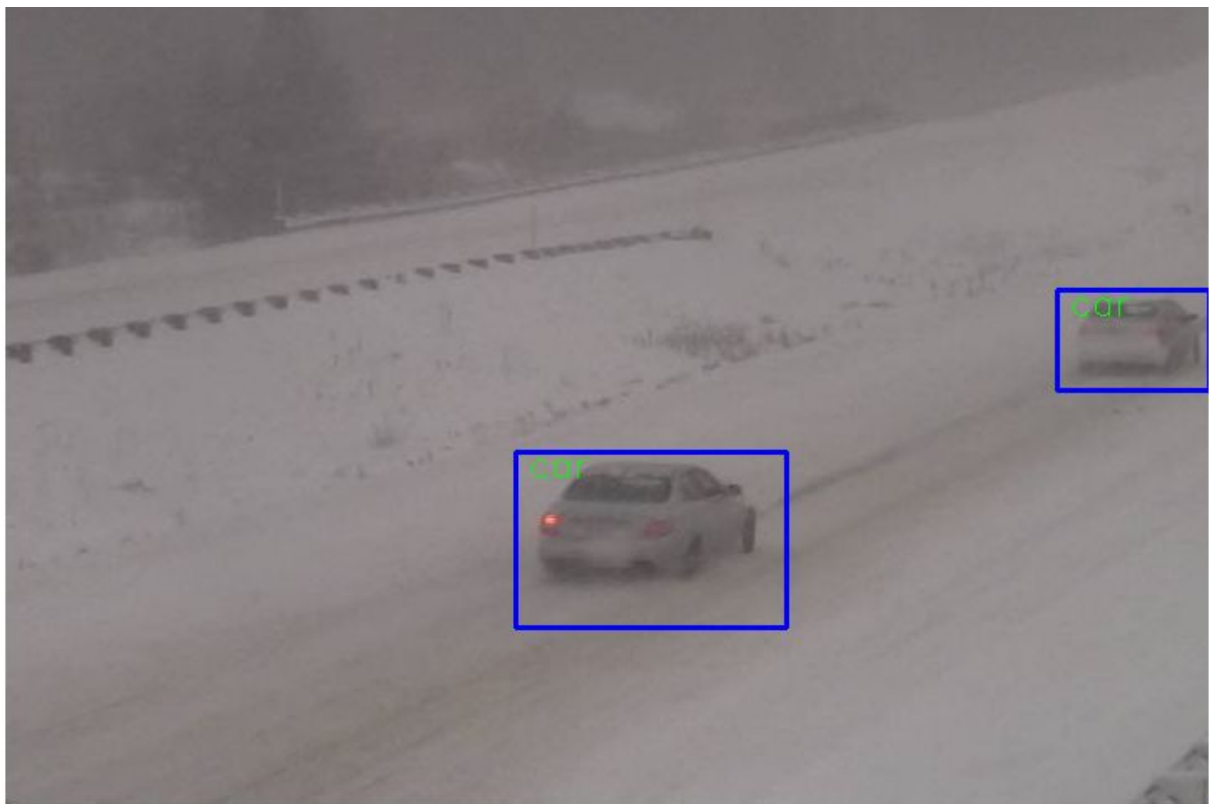
ça donne "human" comme résultat

Test sequence :

Cette partie est divisé dans 3 étapes :

- Appliqué un background subtraction
- Segmenter les objets retrouver en 1
- Pour chaque objets trouver en 2
 - Extraire les points sift
 - Transformer le descripteur de sift trouver en utilisant la PCA calculé précédemment
 - Créé le vecteur d'input
 - Appelle au predict du modèle d'apprentissage
 - Englober l'objet par un rectangle
 - Label l'objet par le nom de class prédit

Exemple :



NOTE : Les résultats sont un peu bruité a cause de résultats de soustraction du fond (comme le MOG de opencv2 n'a pas donner de bon résultats donc

j'ai utilisé mon propre algorithm de gaussien unique
en gros ça donne de bon resultats mais c'est un
peu bruité quand meme)