Projet 8 : Déployez un modèle dans le cloud

Lancelot LECLERCQ

10 juin 2022

Sommaire

Introduction

Problématique

- Préservation de la biodiversité des fruits
  - traitements spécifiques pour chaque espèce de fruits par des robots cueilleurs intelligents

- 1ère étape développer une application mobile
  - Permettre à l'utilisateur d'obtenir des informations sur un fruit à partir d'une photo
  - Sensibiliser le grand public à la biodiversité des fruits
  - Mettre en place une première version du moteur de classification des images de fruits
  - Construire une première version de l'architecture Big Data

Données

- Utilisation du jeu de données Kaggle : `https://www.kaggle.com/datasets/moltean/fruits`
- Nombre total d'images : 90483
- Taille du jeu d'entrainement : 67692 images
- Taille du jeu de test : 22688 images
- Nombre de fruits : 131

- Exemple d'image de fruit :

Services AWS

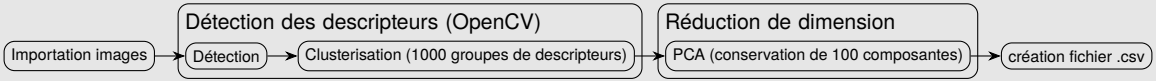
Services AWS

- Utilisation d'une machine EC2 hébergée par Amazon Web Services
  - Utilisation de Debian
  - Installation de java, git
- Utilisation de S3 comme espace de stockage pour la base de données d'images

Chaine de traitement

Spark

- Parallélisation des calculs
  - Utilisation de plusieurs machines
  - Distribution des calculs
- Coût d'utilisation des machines à surveiller



PySpark

- Utilisation de PySpark une interface Spark en python

- Importation des images au format binaire :

```
ImgData = spark.read.format('binaryFile') \
                    .option('pathGlobFilter', '*.jpg') \
                    .option('recursiveFileLookup', 'true') \
                    .load(path) \
                    .select('path', 'content')
```

- Création d'une colonne avec les descripteurs détectés pas OpenCV :

```
udf_image = F.udf(
    get_desc,
    ArrayType(ArrayType(FloatType(), containsNull=False), containsNull=False))

ImgDesc = ImgData.withColumn("descriptors", F.explode(udf_image("content")))
```

- Réduction de dimension par PCA :

```
VA = VectorAssembler(inputCols=BoVW.drop('label').columns,
                     outputCol='features')
pca = PCA(k=100, inputCol='features', outputCol='pca_features')
pipe = Pipeline(stages=[VA, pca])

pipePCA = pipe.fit(BoVW)

pcaData = pipePCA.transform(BoVW)
pcaDataDF = pcaData.select(['label', 'pca_features']).toPandas()
```

Axes d'amélioration

Axes d'amélioration

- KMeans instable, difficile à débugger :

⇒ Essayer d'autres alternatives comme dask

- Utilisation d'un EMR permettant d'avoir les calculs réalisés par plusieurs machines

- Ajout d'un modèle de classification afin de classer les fruits

- Affiner les catégories avec des maturités différentes afin de cueillir les fruits au meilleur moment