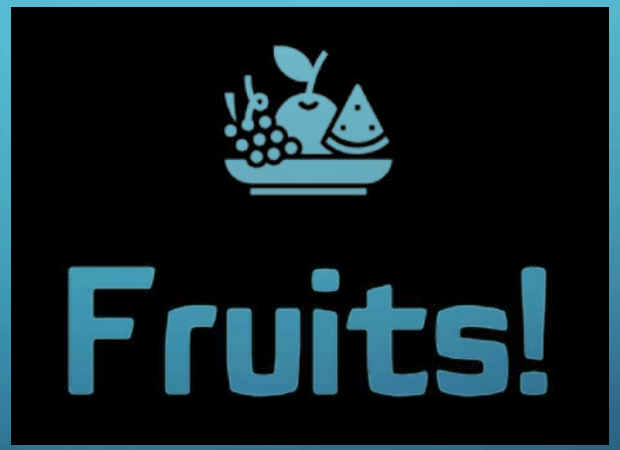
DÉPLOYER UN MODÈLE DANS LE CLOUD



PRÉSENTATION PAR HORTENSE MONNARD

Problématique

Fruits! veut proposer une solution innovante pour la récolte des fruits.

→ Permettre un traitement spécifique pour chaque espèce de fruit, les robots cueilleurs intelligents seront capables de distinguer les espèces de fruit entre elles.

Objectifs

Développer dans un environnement Big Data à l'intérieur duquel on trouve :

- Une première chaîne de traitement des données qui comprend :
  - Le pre-processing ;
  - Une étape de réduction de dimension.

Plan

1. Environnement Big Data :

a. Architecture sur un Cloud

b. Architecture AWS

c. Architecture Spark

d. Pipeline du Projet

2. Analyse des données :

a. Description du Jeu de Données

b. Pre-processing

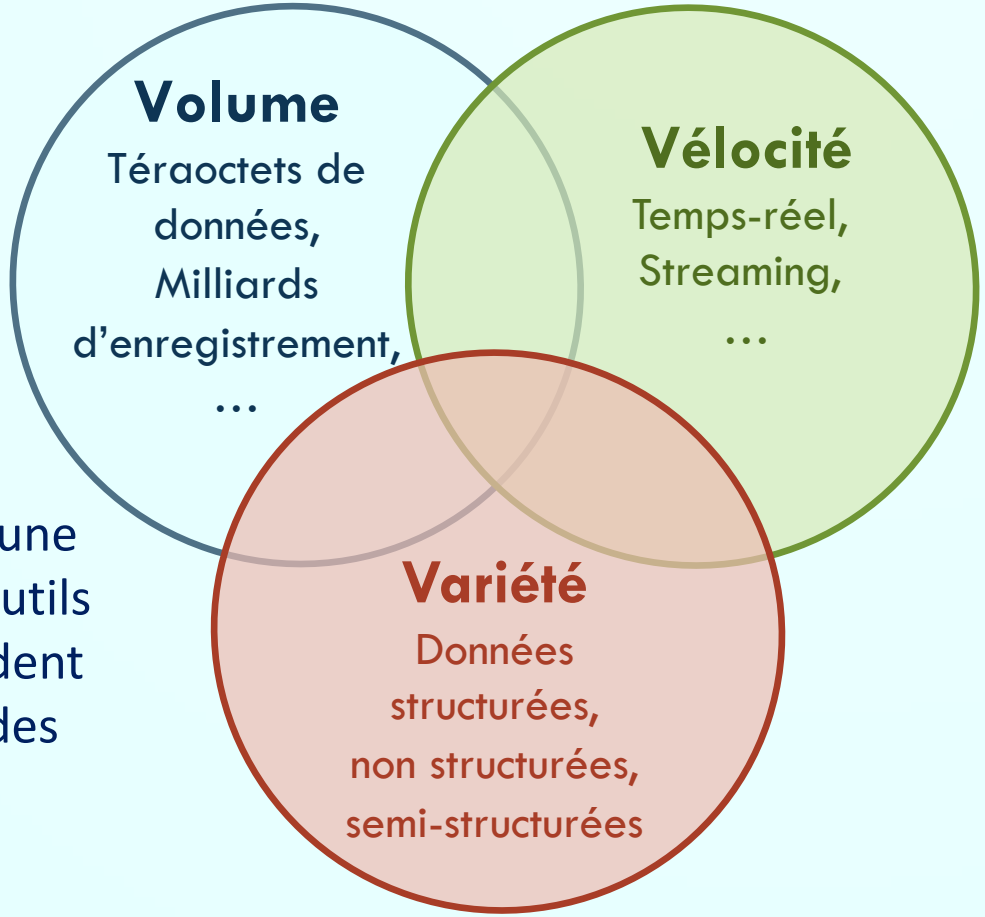
c. Réduction de dimension

1. Environnement Big Data

a. Architecture sur un Cloud

Architecture sur le Cloud

Permet d'accéder en temps réel à des bases de données géantes



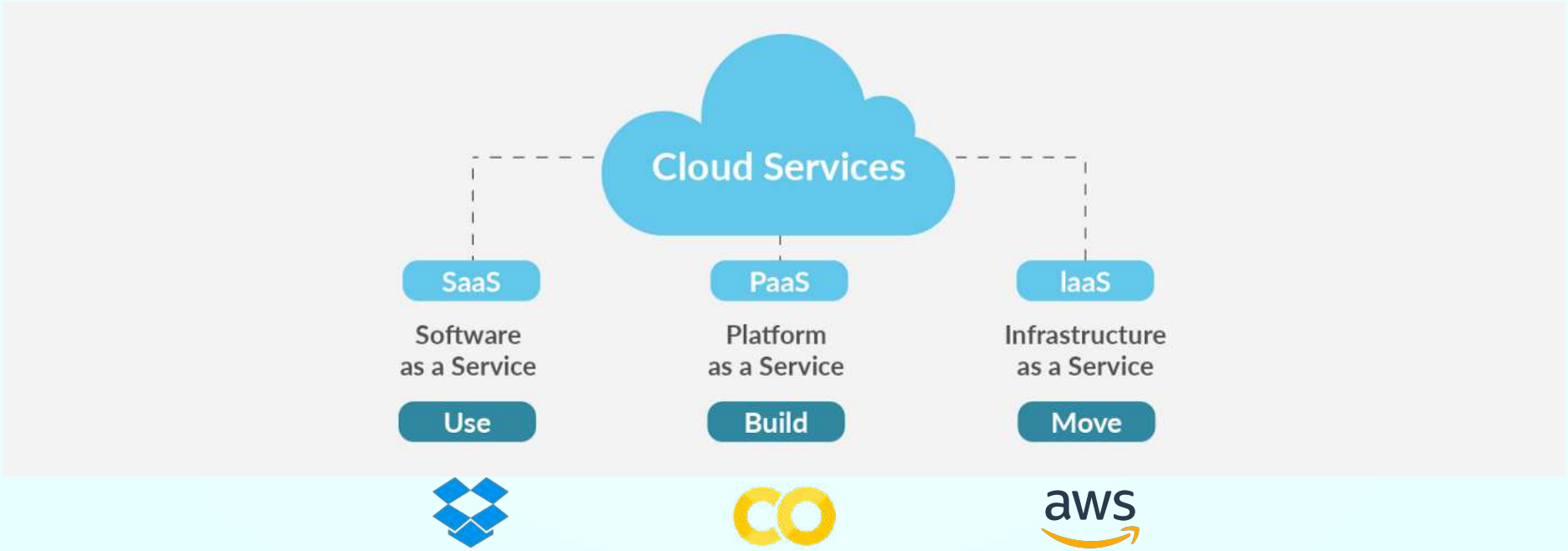
Regroupe une famille d'outils qui répondent à la règle des « 3 V »

3 grands avantages :

- Approvisionnement en libre-service
- Élasticité
- Paiement à l'utilisation

Architecture sur le Cloud

Types de solutions de cloud computing :



Tâches qui peuvent s'effectuer sur le Cloud : machines virtuelles, instances de conteneurs, service d'application, moteur de calcul, stockage de données, etc.

Ici, nous allons utiliser l'IaaS AWS.

b. Architecture AWS

Architecture AWS

AWS Identity and Access Management (IAM)

IAM : Sécurité de connexion
→ Attribution de permissions et de rôles à chaque utilisateur en fonction de ses besoins

Permissions

Temporary security credential

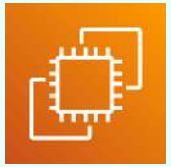
Role

Long-term security credential

Amazon Simple Storage Service (Amazon S3)

S3 : Stockage
→ Stockage dans des buckets sur le cloud (chaque bucket est situé dans une région spécifique)

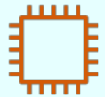
Bucket with objects



Amazon Elastic Compute Cloud (Amazon EC2)

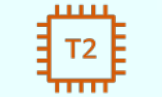
EC2 (ou EMR) : Instance pour le calcul distribué
→ Création d'instances à partir d'images (AMI) afin d'effectuer un calcul distribué sur le cloud dans une région similaire à celle du stockage

AMI



Instance

Architecture AWS



T2 instance

Instance EC2 choisie :

- Type t2.xlarge
- 4 CPU
- 16Go de RAM

Suffisant pour un test sur une partie du jeu de données

Configuration installée :

- Anaconda,

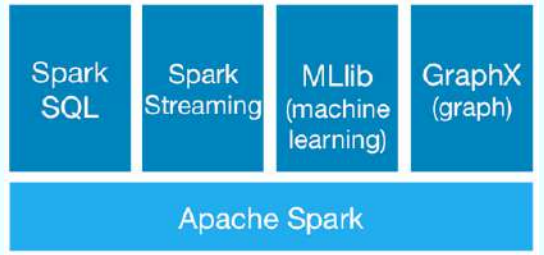
pour python et ses librairies principales

- Spark
- Java
- Tensorflow
- Findspark

c. Architecture Spark

Architecture Spark

Spark est un **framework de calcul distribué** pour le traitement et l'analyse de **données massives**.
Il peut travailler sur des données sur disque ou des **données chargées en RAM** (in memory), sans perte de performance.



Spark est une plate-forme riche en fonctionnalités : **machine learning (Mllib)**, SQL, streaming, graph.

Pour distribuer les calculs, Spark a besoin **d'exécuteurs** ou machines de calcul (comme EC2).

Les exécuteurs exécutent du code Spark.
Cependant, le conducteur peut être « piloté » à partir d'un certain nombre de langues différentes par l'intermédiaire des API de différents langages.

Architecture Spark

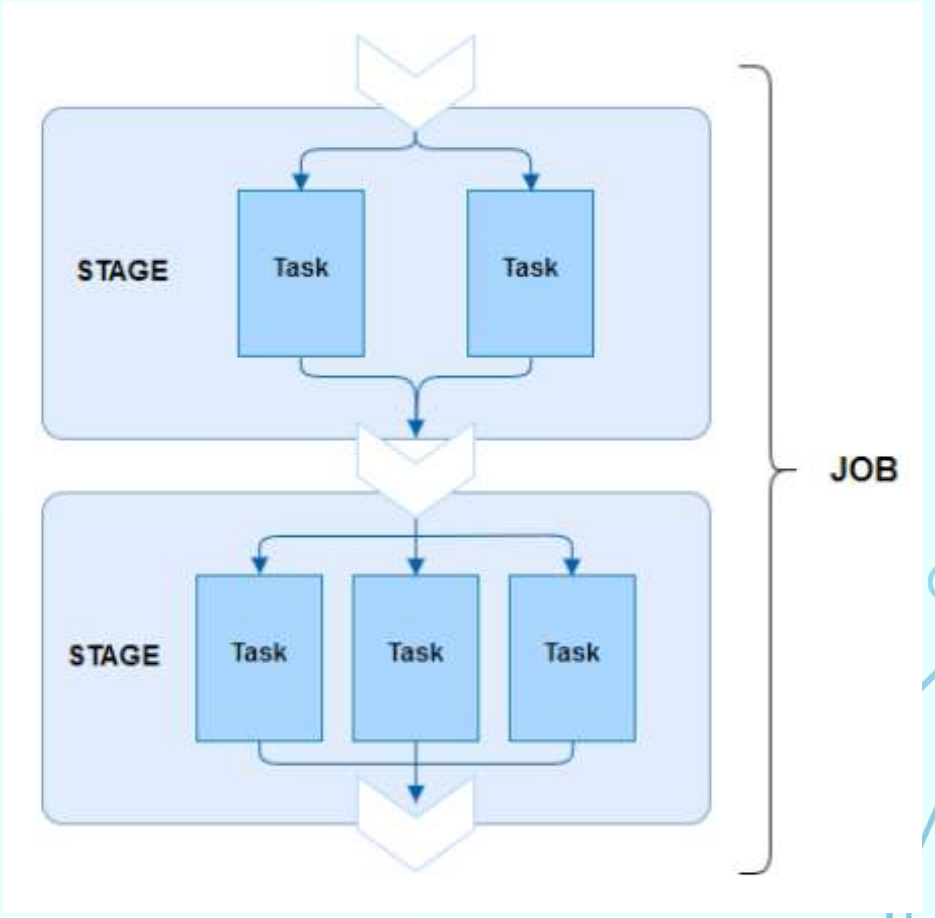
Plusieurs **types d'opérations** sont possibles sur Spark :

- **Transformations** : ne renvoient aucun résultat ;
- **Actions** : déclenchent un calcul (ex : .count()).

Un **job** correspond à une action sur un RDD et est composé de plusieurs **étapes** (stages) séparées par des shuffles.

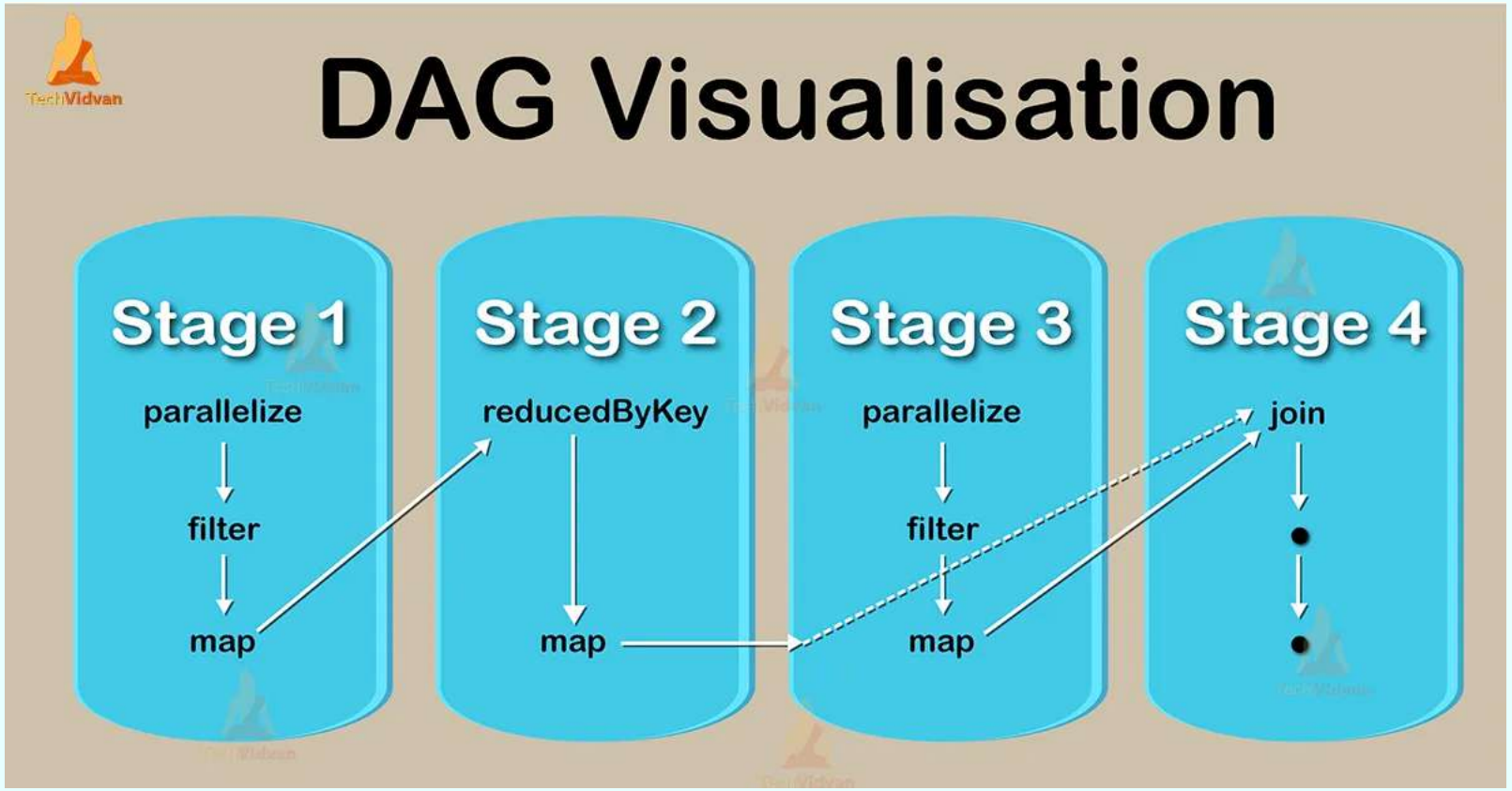
Une **étape** correspond à un ensemble de **tâches** réalisées en parallèle.
Il ne peut y avoir qu'une action par étape.

Il existe une **tâche** par opération et par partition.



Architecture Spark

Graphique Acyclique Orienté :

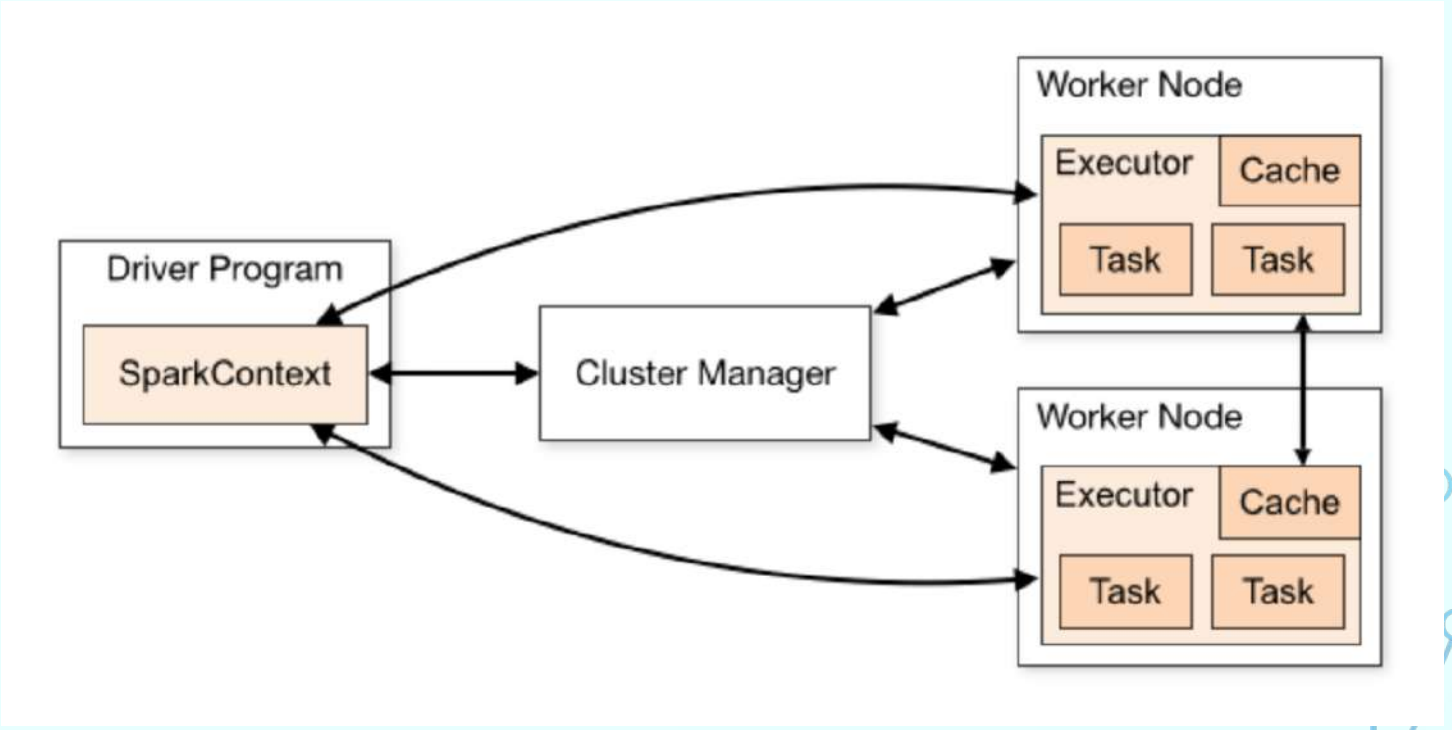


Le DAG est un graphique qui récapitule les opérations appliqués sur un RDD.

Architecture Spark

Les applications Spark sont exécutées comme des **processus indépendants sur un cluster** (ou grappe de serveurs), coordonnés par l'objet « **SparkContext** » (contenu dans « **SparkSession** ») dans le programme de pilotage (driver program).

- Spark emploie un **cluster manager qui assure le suivi des ressources disponibles**.
- Le **programme de pilotage est responsable de l'exécution le programme à travers les exécuteurs** pour accomplir une tâche donnée.



https://spark.apache.org/docs/latest/cluster-overview.html

Architecture Spark

Quelques fonctions gérées par SparkContext/ SparkSession :

- Retourner le statut actuel de l'application
- Paramétrer la configuration
- Annuler un Job
- Annuler une Etape
- Clôturer la session spark
- Créer des RDDs persistants ou non persistants
- Accéder aux RDDs persistants
- Accéder à différents services (AWS security credentials, pour se connecter à un bucket)
- Allocation dynamique d'exécuteurs pour un « *elastic scaling* »

c. Pipeline du Projet

Fruits!

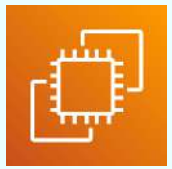
Pipeline du Projet

1

Amazon Simple Storage Service (Amazon S3)

Stockage des données sur aws S3

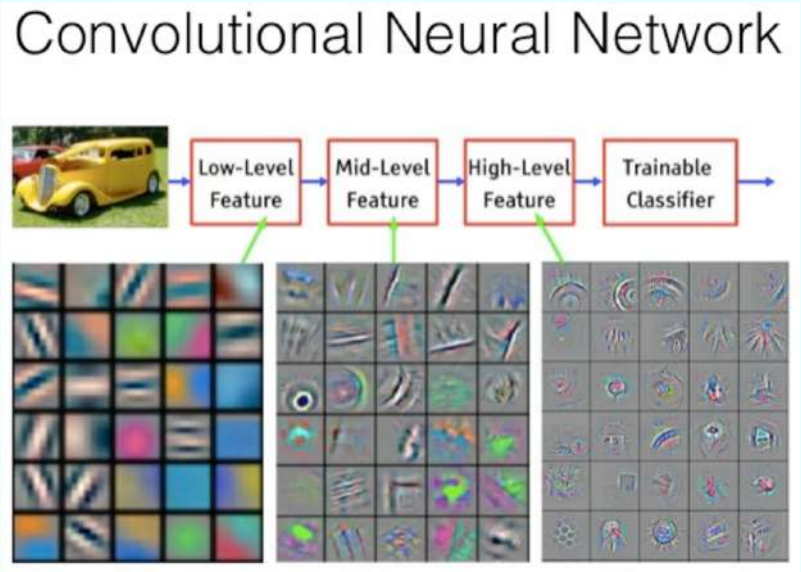
2



Amazon Elastic Compute Cloud (Amazon EC2)

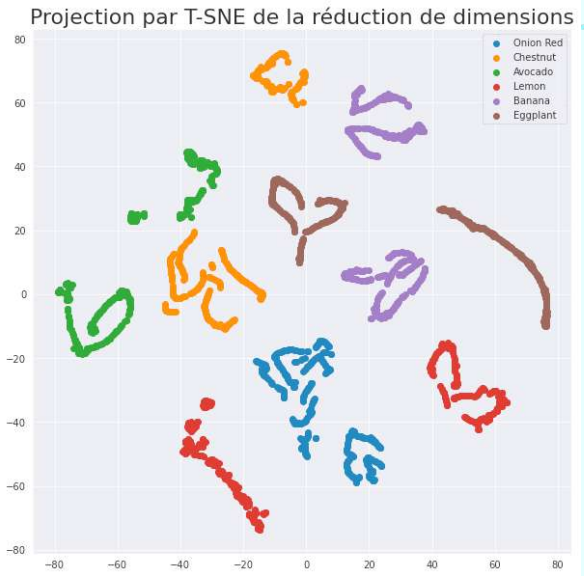
Chargement des données en mémoire sur aws EC2

3



Extraction de features grâce à ResNet50, via un calcul distribué avec Spark

4



Réduction de dimensions par PCA, via un calcul distribué avec Spark

2. Analyse des données

a. Description du Jeu de Données

Fruits!

Description du Jeu de Données

Test sur un Jeu Réduit :
2777 images et 6 catégories

Jeu de Données Entier :
90 483 images et 131 catégories

2 Jeux de Données Etiquetées :
90 380 images et 131 catégories

+

1 Jeu de Test Non- Etiquetées :
103 images

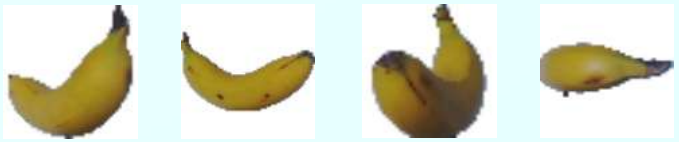
1 Jeu d'Entrainement :
67 692 images et 131 catégories

+

1 Jeu de Test :
22 688 images et 131 catégories

Images avec étiquettes

→ **Apprentissage Supervisé**



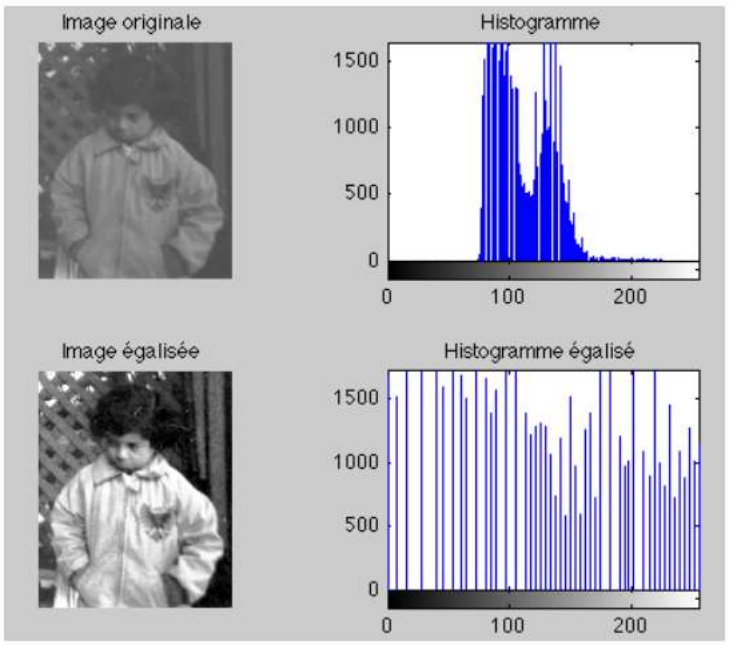
Multiples images sous différents angles pour chaque fruit

b. Pre-Processing

Pre-Processing

Pré-traitement des images

1. Egalisation des histogrammes



2. Application d'un filtre gaussien

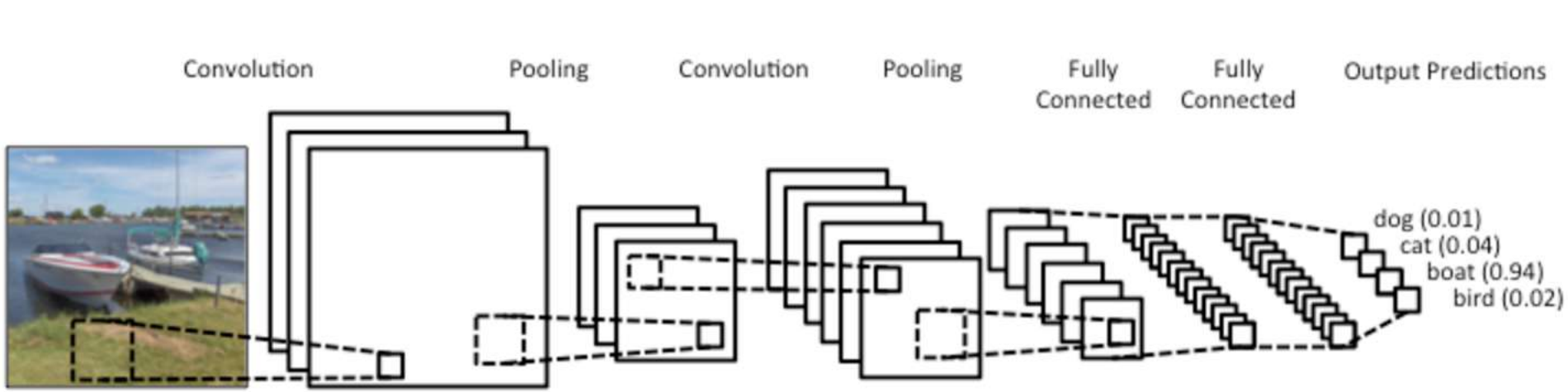
avant

après

Pre-Processing

Extraction de features

Deep Learning avec Réseau de Neurones Convolutif (CNN) de type ResNet50



Les **filtres de convolution** permettent une **extraction de features** qui facilite la **classification**.

c. Réduction de Dimensions

Réduction de Dimensions

ResNet produit une représentation des images avec 2048 features (ou vecteurs dimensionnels).

La réduction de dimension permet de sélectionner les features permettant d'expliquer la variance.

Réduction de dimension par PCA

```
1 # Apply a PCA on the features extracted by the CNN
2 # Determine the optimal number of features
3 df_pca_opt = pca_optimisation(df_features, n_components=2048)
```

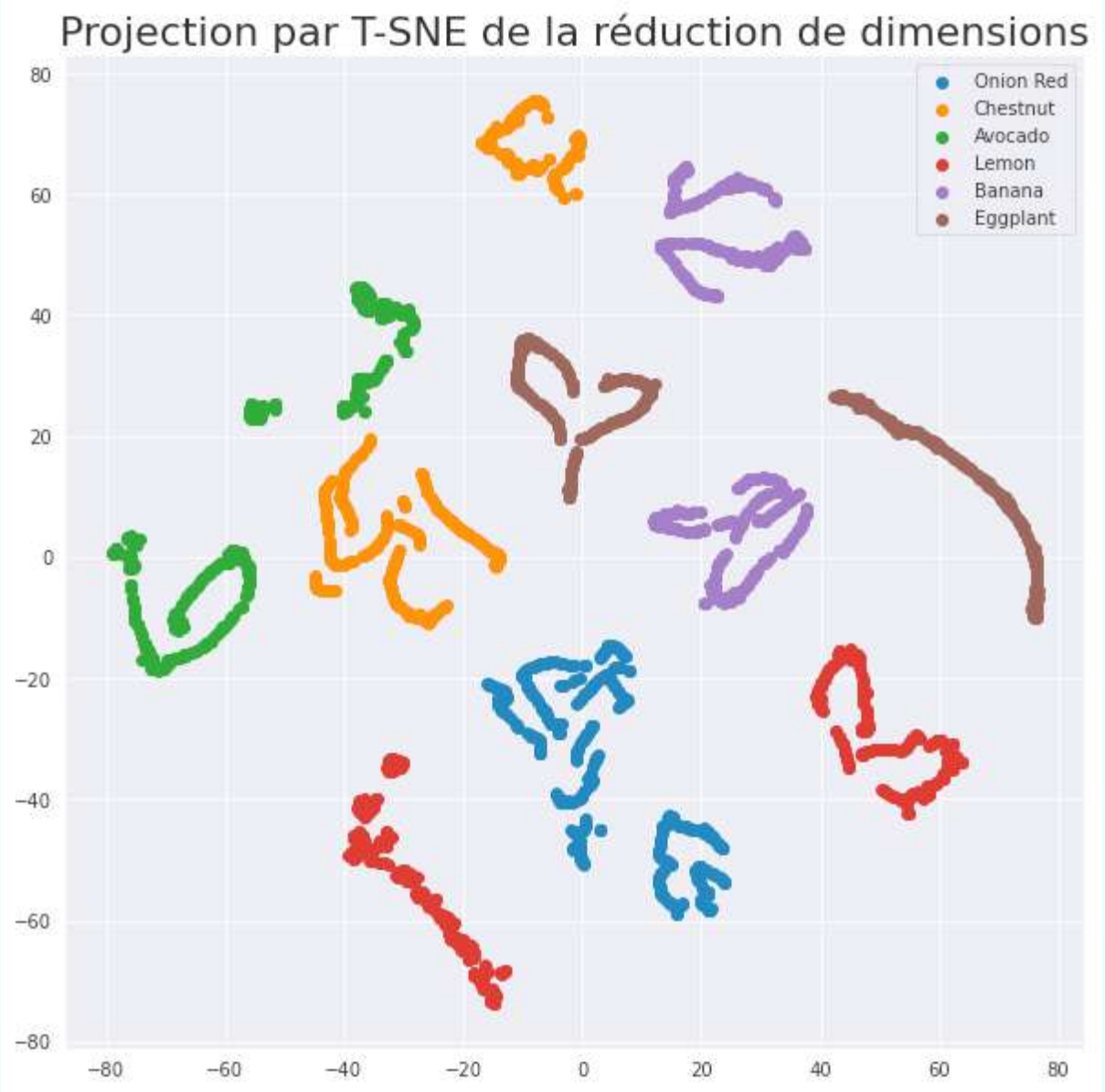
```
Temps d'entrainement du modèle de PCA 49.27 secondes
96 composantes principales expliquent au moins 95% de la variance totale
```

```
1 # Apply a PCA with the optimal number of components
2 df_pca = pca_transformation(df_features, n_components=96)
```

```
Temps d'entrainement du modèle de PCA 46.01 secondes
```

Réduction de Dimensions

Représentation sur 2 dimensions des 96 features extraites grâce à l'analyse par composante principale (PCA)



3. Conclusion et Perspectives

a. Conclusion

Conclusions

- La réduction de dimension est une étape essentielle :

On passe de 2048 features à 96 features.

- Le temps de calcul peut être long et le test à été fait sur un échantillon du jeu données « Training ».

- L'architecture est particulièrement importante pour un bon fonctionnement de Spark.

b. Perspectives

Perspectives

- Améliorer prétraitement : application d'un filtre de Gabor
- Modéliser la classification : Random Forest ou un autre CNN.
- Utilisation d'un cluster EMR :
  - Environnement Spark déjà préinstallé ;
  - Meilleures capacités grâce à l'utilisation de plusieurs instances pour le calcul distribué.
- Utilisation de GPU :
  - Meilleures performances de calcul.
- Amélioration de la détection, via une plus grande base de données :
  - Détection des fruits inaptes à la consommation (maladies, parasites).

MERCI
POUR VOTRE ATTENTION

ANNEXES

Lancement de Spark

Lancer la session Spark

```
# Configure AWS security confidentials
endpoint = "s3.eu-west-1.amazonaws.com"
access_key = "AKIA2QP5P4KL7CR4HHWC"
secret_key = "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"
```

```
# Configure SparkContext
sc = SparkContext()

sc._jsc.hadoopConfiguration().set('my.mapreduce.setting', 'someVal')
sc._jsc.hadoopConfiguration().set("fs.s3a.endpoint", endpoint) #point
sc._jsc.hadoopConfiguration().set("fs.s3a.access.key", access_key)
sc._jsc.hadoopConfiguration().set("fs.s3a.secret.key", secret_key)
```

```
# Creating SparkSession
spark = (SparkSession
         .builder
         .master("local[*]")
         .appName('Projet8_OC')
         .getOrCreate()
)

print('La session Spark a été lancé avec succès')
```

```
La session Spark a été lancé avec succès
```

```
# Print informations about the session
spark
```

SparkSession - in-memory
SparkContext

Spark UI
Version
`v3.0.2`
Master
`local[*]`
AppName
`pyspark-shell`

Chargement des données

```
# Apply the function to create a dataframe
create_images_dataframe(input_data)
```

```
Les images ont été chargées
```

```
# Print the number of images in the dataframe
df_tot.count()
```

```
2777
```

Créer une feature avec les catégories

```
# Define a new dataframe with the categories
df_tot = df_tot.withColumn('category', element_at(split(df_tot['path'], '/'), -2))
```

```
# Print the schema of the dataframe
df_tot.printSchema()
```

```
root
 |-- path: string (nullable = true)
 |-- modificationTime: timestamp (nullable = true)
 |-- length: long (nullable = true)
 |-- content: binary (nullable = true)
 |-- category: string (nullable = true)
```

```
# Print the dataframe
df_tot.show(5)
```

```
+--------------------+-------------------+------+--------------------+---------+
|                path|   modificationTime|length|             content| category|
+--------------------+-------------------+------+--------------------+---------+
|s3a://hortensebuc...|2021-09-01 06:22:45|  6233|[FF D8 FF E0 00 1...|Onion Red|
|s3a://hortensebuc...|2021-09-01 06:22:45|  6230|[FF D8 FF E0 00 1...|Onion Red|
|s3a://hortensebuc...|2021-09-01 06:22:45|  6209|[FF D8 FF E0 00 1...|Onion Red|
|s3a://hortensebuc...|2021-09-01 06:22:45|  6209|[FF D8 FF E0 00 1...|Onion Red|
|s3a://hortensebuc...|2021-09-01 06:22:45|  6201|[FF D8 FF E0 00 1...|Onion Red|
+--------------------+-------------------+------+--------------------+---------+
only showing top 5 rows
```

```
# Print unique values for 'category'
df_tot.select('category').distinct().show()
```

```
+---------+
| category|
+---------+
|   Banana|
|    Lemon|
|  Avocado|
| Chestnut|
|Onion Red|
| Eggplant|
+---------+
```

Modèle du CNN ResNet 50 utilisé

```python
# Model
model = ResNet50(
        include_top=False, # delete top layer
        weights=None,
        input_shape=(100,100,3),
        pooling='avg'
)
```

```python
model.summary()
```

```
_____
conv5_block3_3_bn (BatchNormali (None, 4, 4, 2048)   8192        conv5_block3_3_conv[0][0]
_____
_____
conv5_block3_add (Add)          (None, 4, 4, 2048)   0           conv5_block2_out[0][0]
                                                                 conv5_block3_3_bn[0][0]
_____
_____
conv5_block3_out (Activation)   (None, 4, 4, 2048)   0           conv5_block3_add[0][0]
_____
_____
avg_pool (GlobalAveragePooling2 (None, 2048)         0           conv5_block3_out[0][0]
==========================================================================================
=======
Total params: 23,587,712
Trainable params: 23,534,592
Non-trainable params: 53,120
_____
```

Extraction de features

```
# Run the featurization on the dataframe
# NB: It can take some time
df_features = df_tot.repartition(16).select(col('path'),
                                            col('category'),
                                            featurize_udf('content').alias('features'),
                                            )
```

```
df_features.count()
```

```
2777
```

```
# Make a transient instance persistent
df_features.persist()
```

```
DataFrame[path: string, category: string, features: array<float>]
```

```
df_features.show(5)
```

```
+--------------------+---------+--------------------+
|                path|  category|            features|
+--------------------+---------+--------------------+
|s3a://hortensebuc...|Onion Red|[3.0996382, 8.971...|
|s3a://hortensebuc...|Onion Red|[2.951924, 9.1819...|
|s3a://hortensebuc...|Onion Red|[2.9499738, 9.040...|
|s3a://hortensebuc...|Onion Red|[3.1436825, 9.163...|
|s3a://hortensebuc...|Onion Red|[3.1653037, 9.675...|
+--------------------+---------+--------------------+
only showing top 5 rows
```