Projet de Data Science (Openclassrooms PJ7)
Réaliser des inexations automatiques d’images

Dogs
dataset

10/2020

François BOYER

L’enjeu

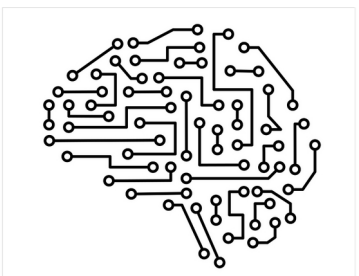
Réaliser un modèle capable de **prédire la race d’un chien** à partir d’une photo

Difficultés :
- 120 races différentes dont certaines se ressemblent beaucoup
- Forte variation intra classe des images: âge, pose, couleur, présence d’humains sur les photos et présence d’un décor de fond

1ère approche
Machine learning classique

Algo. d’extraction de features et classifieur

2ème approche
Deep learning



Réseau neuronal artificiel

Première partie du projet

Approche machine learning classique

Démarche suivie

Formulation de l’objectif

Implémentation d’extraction SIFT avec clustering

Classifieur – 2 classes + tuning

Classifieur – 20 classes + tuning

Formulation de l’objectif

Approche supervisée

Multi classes (120 races dans le dataset)

1 classe par image

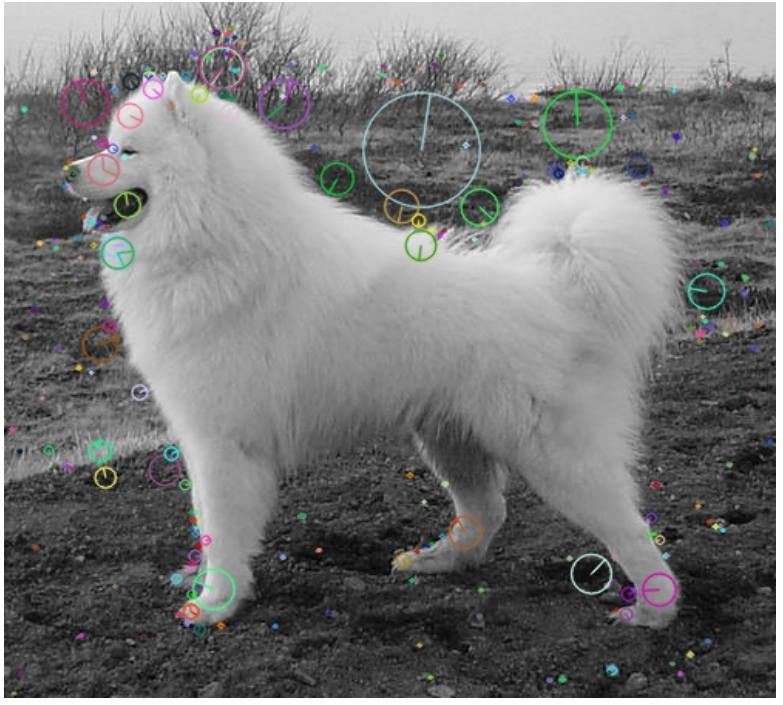
On mesurera les scores
de **precision** et de **rappel** micro

Implémentation d’extraction SIFT avec clustering (implémentation OpenCV)

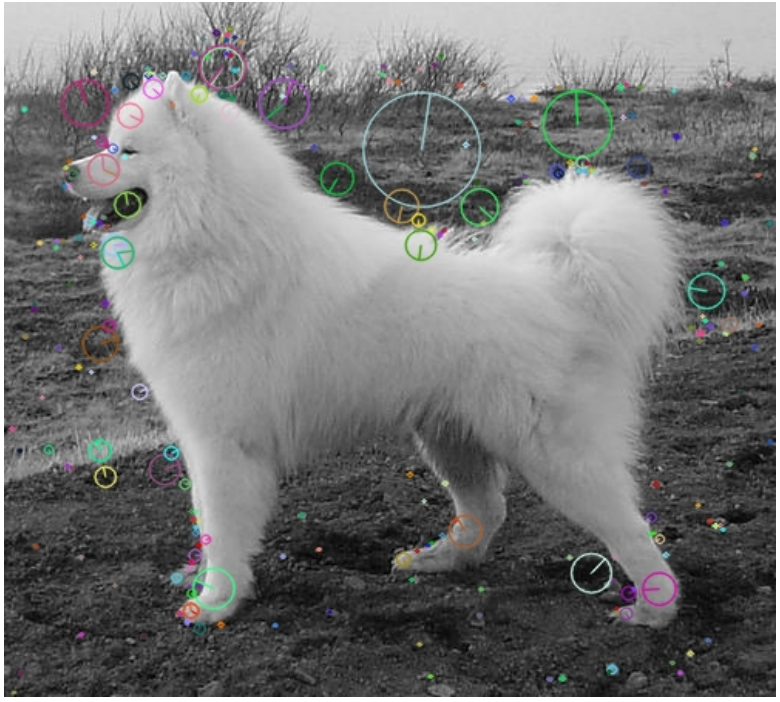
Extraction des keypoints

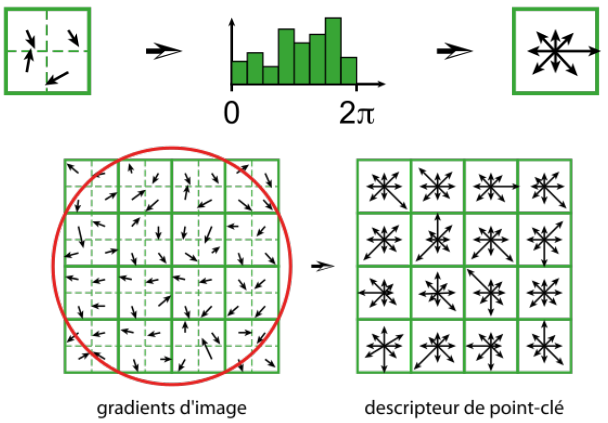
Extraction des keypoints de chaque image

Conservation des P keypoints les plus importants



Calcul des descripteurs





8 histogrammes d'orientation / amplitude de gradient

x

16 descripteurs

Pour chaque point clé :
calcul de 16 x 8 = **128 descripteurs SIFT**

Matrice des descripteurs

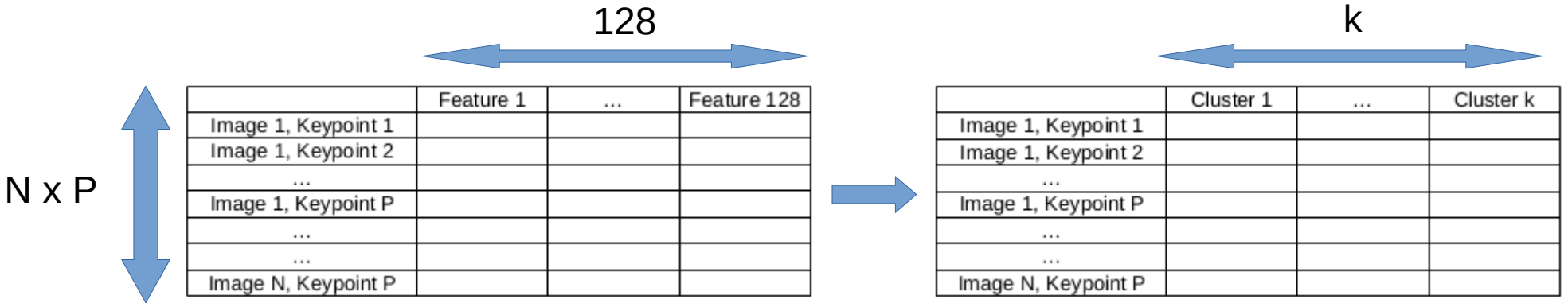
N : nombre d'images = Nombre de chiens par classe **x** Nombre de classes
P : nombre de keypoints par image

N x P

	Feature 1	…	Feature 128
Image 1, Keypoint 1			
Image 1, Keypoint 2			
…			
Image 1, Keypoint P			
…			
…			
Image N, Keypoint P			

Clustering de la matrice des descripteurs

N : nombre d'images = Nombre de chiens par classe **x** Nombre de classes
P : nombre de keypoints par image



Agrégation de la matrice des descripteurs

N nombre d'images = 3
P nombre de keypoints = 2
k nombre de clusters = 3

N x P

	Cluster 1	Cluster 2	Cluster 3
Image 1, Keypoint 1	1	0	0
Image 1, Keypoint 2	0	0	1
Image 2, Keypoint 1	1	0	0
Image 2, Keypoint 2	1	0	0
Image 3, Keypoint 1	0	1	0
Image 3, Keypoint 2	0	0	1

k

N

	Cluster 1	Cluster 2	Cluster 3
Image 1	1	0	1
Image 2	2	0	0
Image 3	0	1	1

Normalisation de la matrice des descripteurs

N nombre d’images = 3
P nombre de keypoints = 2
k nombre de clusters = 3

k

N

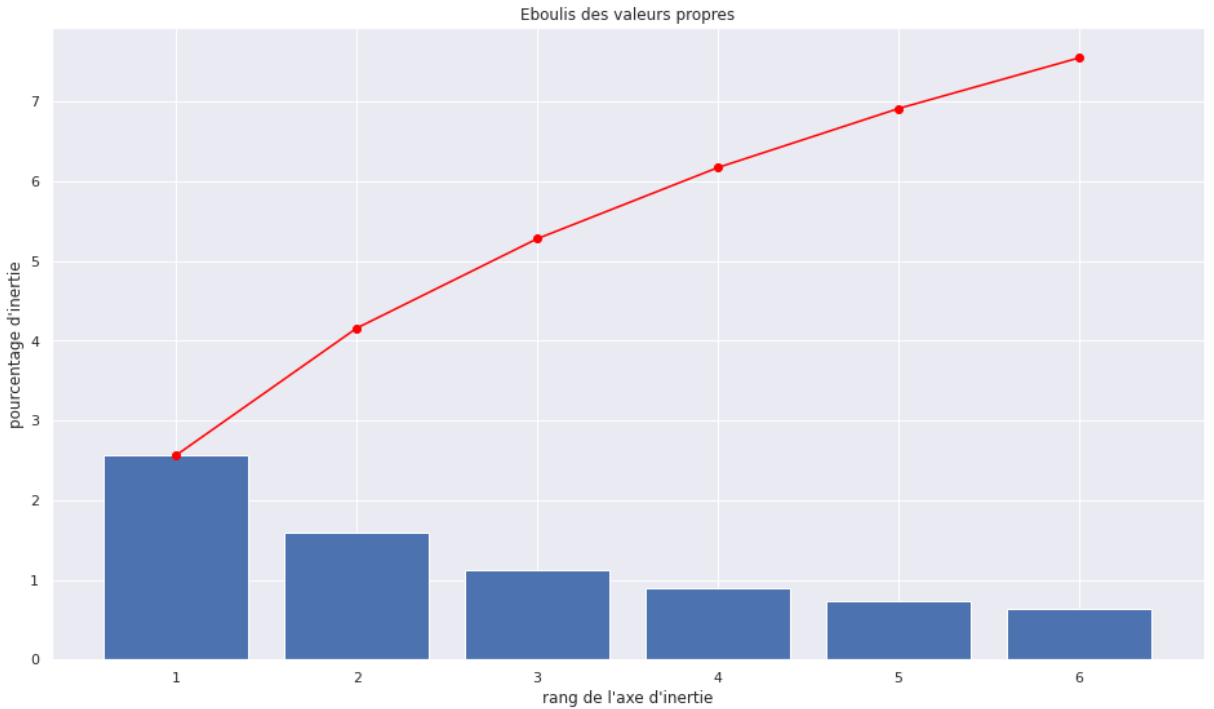
	Cluster 1	Cluster 2	Cluster 3
Image 1	1	0	1
Image 2	2	0	0
Image 3	0	1	1

	Cluster 1	Cluster 2	Cluster 3
Image 1	1	0	1
Image 2	1	0	0
Image 3	0	0.5	0.5

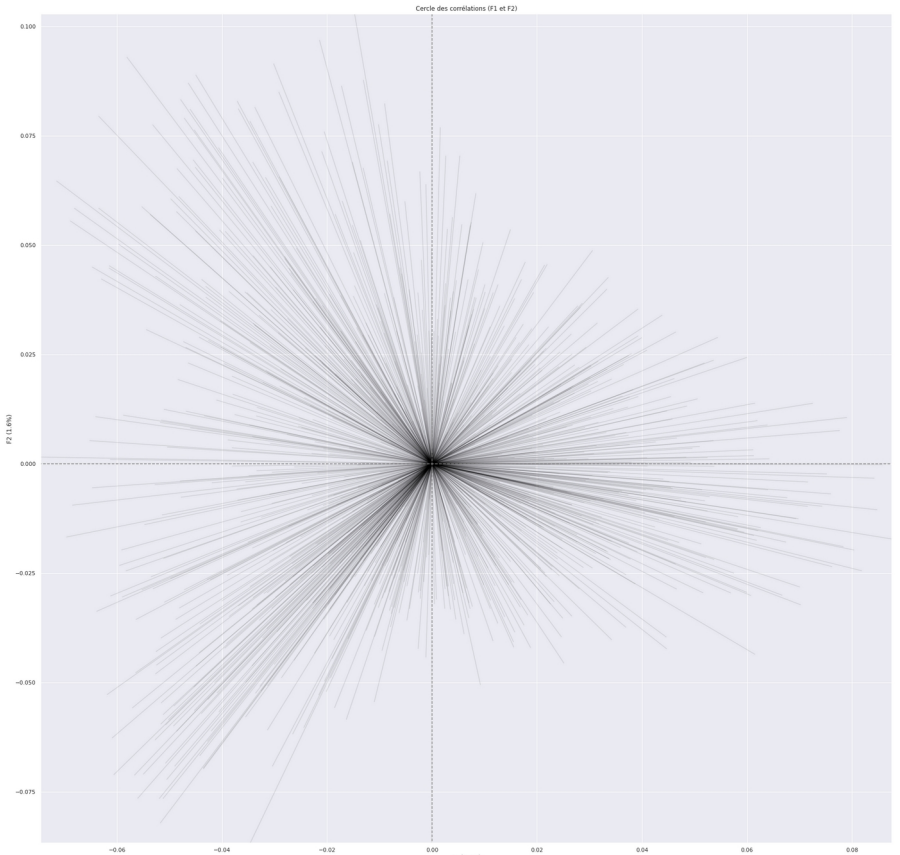
Cette matrice servira d’entrée au classifieur

Analyse en composantes principales

Réduction de dimension avec PCA

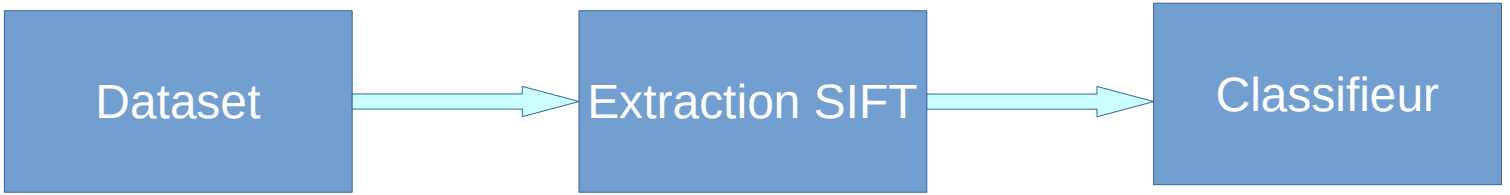


Cercle des corrélations



Implémentation d’un modèle de classification :

Extractions SIFT + classifieur

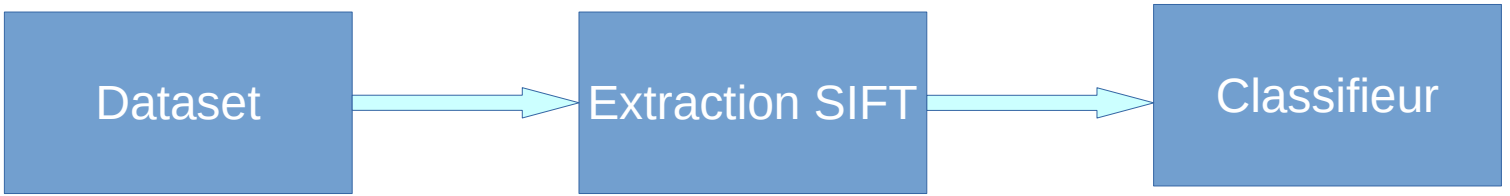


Choix à faire :
Nombre de classes
Nombre de chiens / classe

Choix à faire :
Nombre de KP / image
Nombre de clusters

Choix à faire :
Decision Tree
Random Forest

1ère itération



2 classes
100 chiens / classe

200 KP / image
100 clusters

Decision Tree
Random Forest

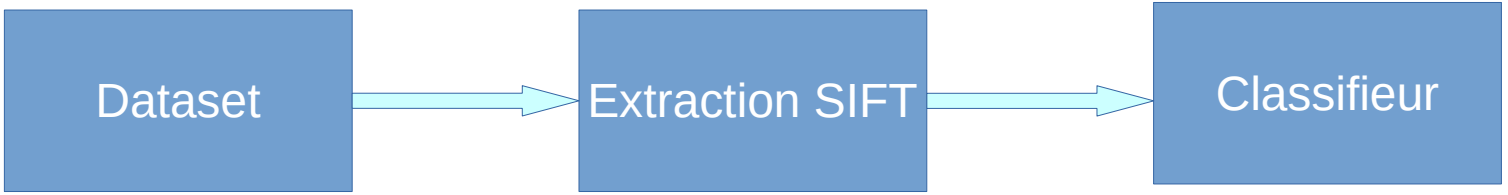
Precision micro et recall micro :
⇒ training set :1
⇒ test set : 0.45 (avec DecisionTree classifier)
0.55 (avec Random Forest après GridSearch)

```
max_depth=10, max_features=20, max_leaf_nodes=50,
n_estimators=100
```

Les résultats ne sont pas meilleurs qu'un classifieur random (0.5)

2ème itération

En jaune ce qui a changé par rapport à l'itération précédente



2 classes
100 chiens / classe

200 KP / image
1000 clusters

Decision Tree
Random Forest

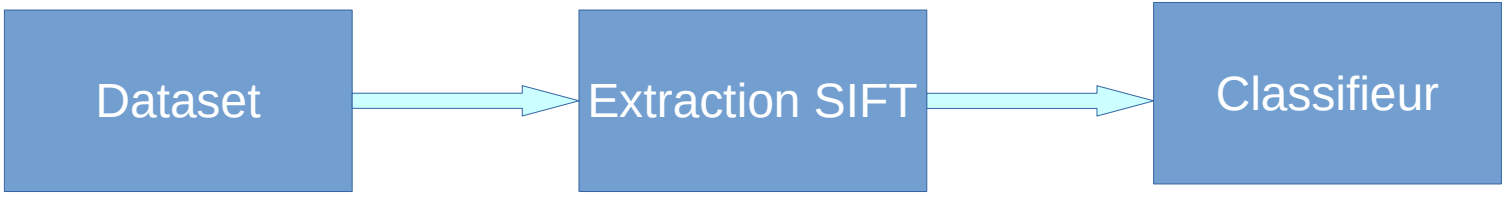
Precision micro et recall micro :
⇒ training set :1
⇒ test set : 0.75 (avec DecisionTree classifier)
0.75 (avec Random Forest)

```
max_depth=10, max_features=20, max_leaf_nodes=50,
n_estimators=100
```

⇒ Les résultats sont meilleurs

3ème itération

En jaune ce qui a changé par rapport à l'itération précédente



20 classes
100 chiens / classe

200 KP / image
1000 clusters

Decision Tree
Random Forest

Precision micro et recall micro :

⇒ training set : 0.99

⇒ test set : 0.03 (avec DecisionTree classifier)

0.125 (avec Random Forest après GridSearch)

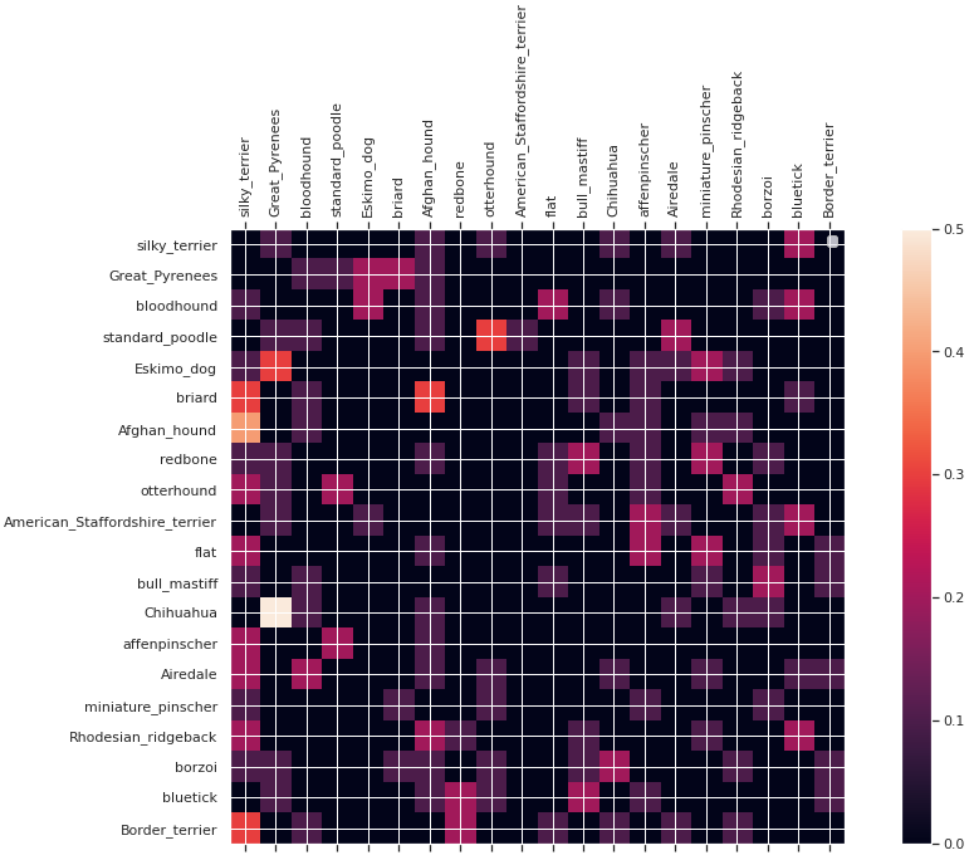
```
max_depth=100, max_features=20,
max_leaf_nodes=50, n_estimators=5000
```

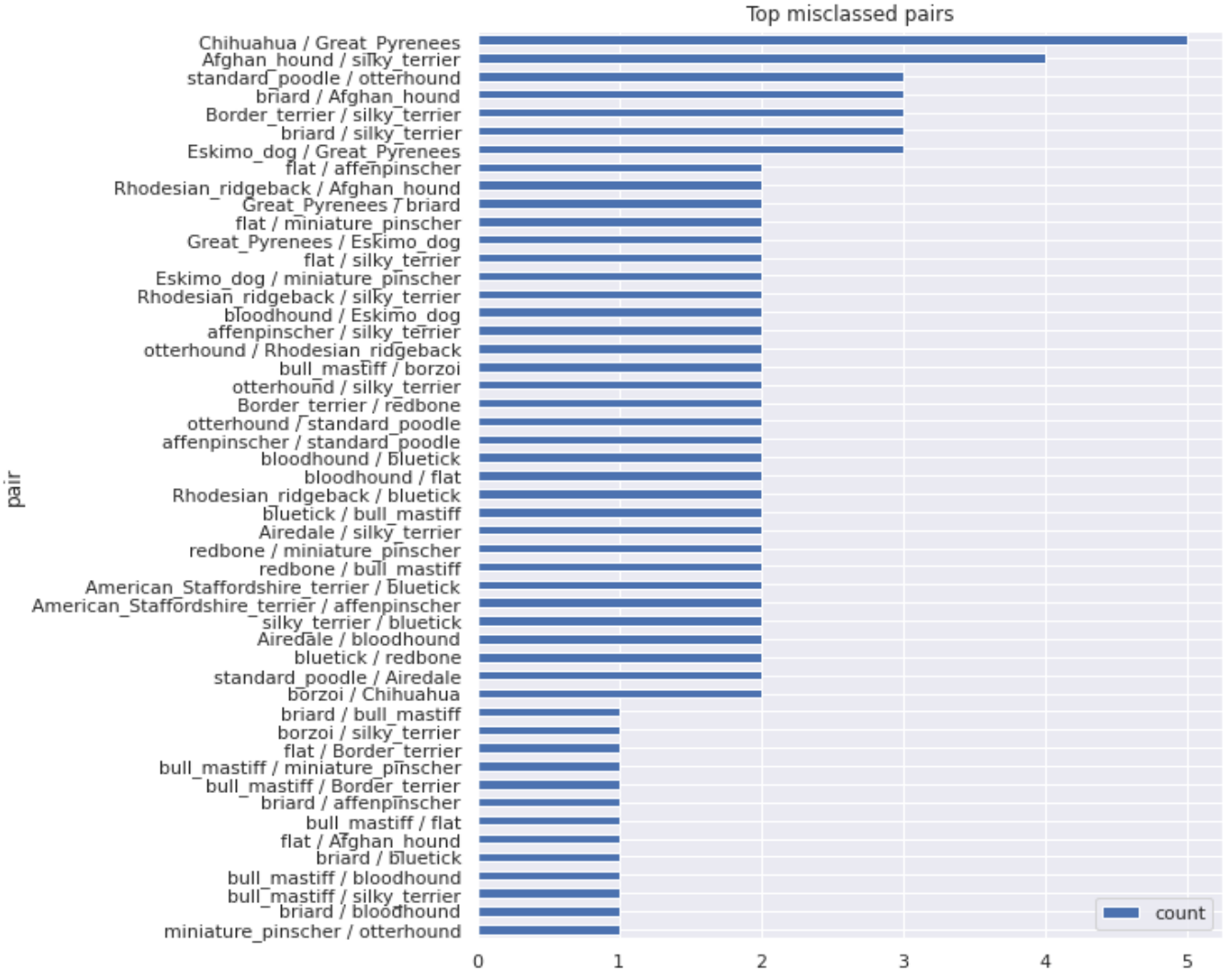
⇒ random : 0,045

Les résultats sont 3 fois mieux qu'avec des features random

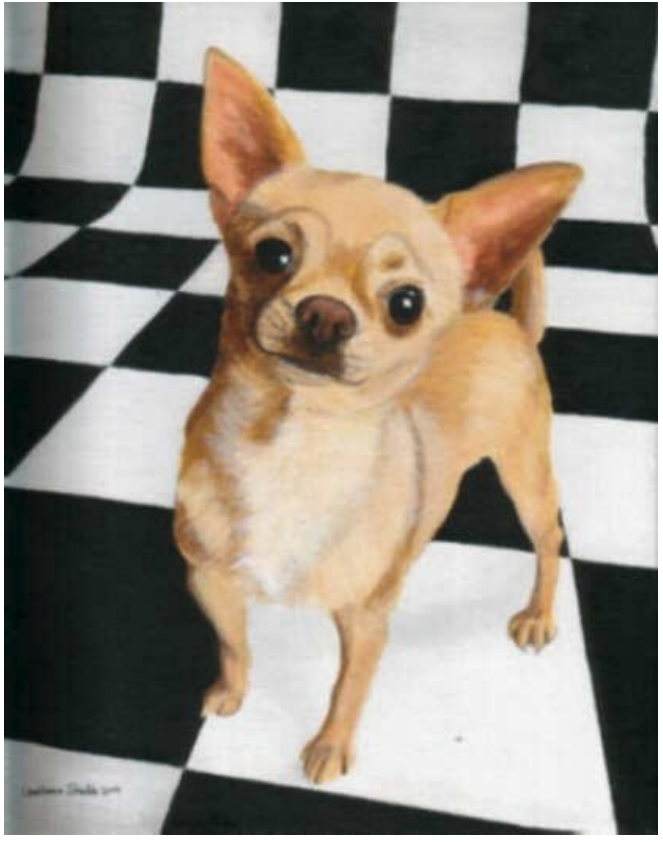
Analyse des erreurs du modèle

Matrice de confusion




Top misclassed pairs
pair
Chihuahua / Great_Pyrenees
Afghan_hound / silky_terrier
standard_poodle / otterhound
briard / Afghan_hound
Border_terrier / silky_terrier
briard / silky_terrier
Eskimo_dog / Great_Pyrenees
flat / affenpinscher
Rhodesian_ridgeback / Afghan_hound
Great_Pyrenees / briard
flat / miniature_pinscher
Great_Pyrenees / Eskimo_dog
flat / silky_terrier
Eskimo_dog / miniature_pinscher
Rhodesian_ridgeback / silky_terrier
bloodhound / Eskimo_dog
affenpinscher / silky_terrier
otterhound / Rhodesian_ridgeback
bull_mastiff / borzoi
otterhound / silky_terrier
Border_terrier / redbone
otterhound / standard_poodle
affenpinscher / standard_poodle
bloodhound / bluetick
bloodhound / flat
Rhodesian_ridgeback / bluetick
bluetick / bull_mastiff
Airedale / silky_terrier
redbone / miniature_pinscher
redbone / bull_mastiff
American_Staffordshire_terrier / bluetick
American_Staffordshire_terrier / affenpinscher
silky_terrier / bluetick
Airedale / bloodhound
bluetick / redbone
standard_poodle / Airedale
borzoi / Chihuahua
briard / bull_mastiff
borzoi / silky_terrier
flat / Border_terrier
bull_mastiff / miniature_pinscher
bull_mastiff / Border_terrier
briard / affenpinscher
bull_mastiff / flat
flat / Afghan_hound
briard / bluetick
bull_mastiff / bloodhound
bull_mastiff / silky_terrier
briard / bloodhound
miniature_pinscher / otterhound
0
1
2
3
4
5
count

Chihuahua



Great Pyrenees

Afghan hound

Silky Terrier

Standard poodle

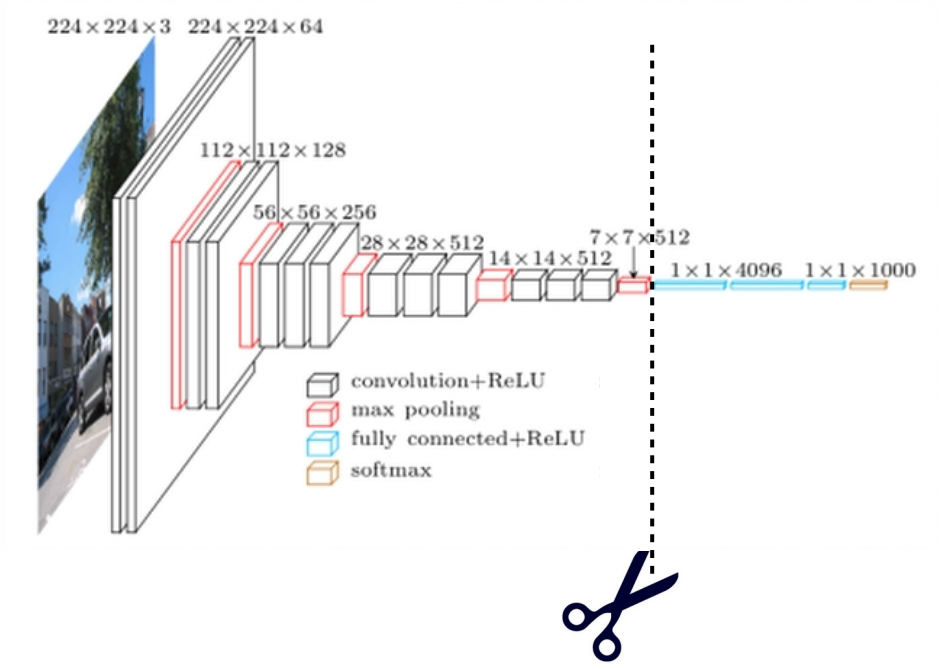
Otterhound

Deuxième partie du projet
Approche deep learning

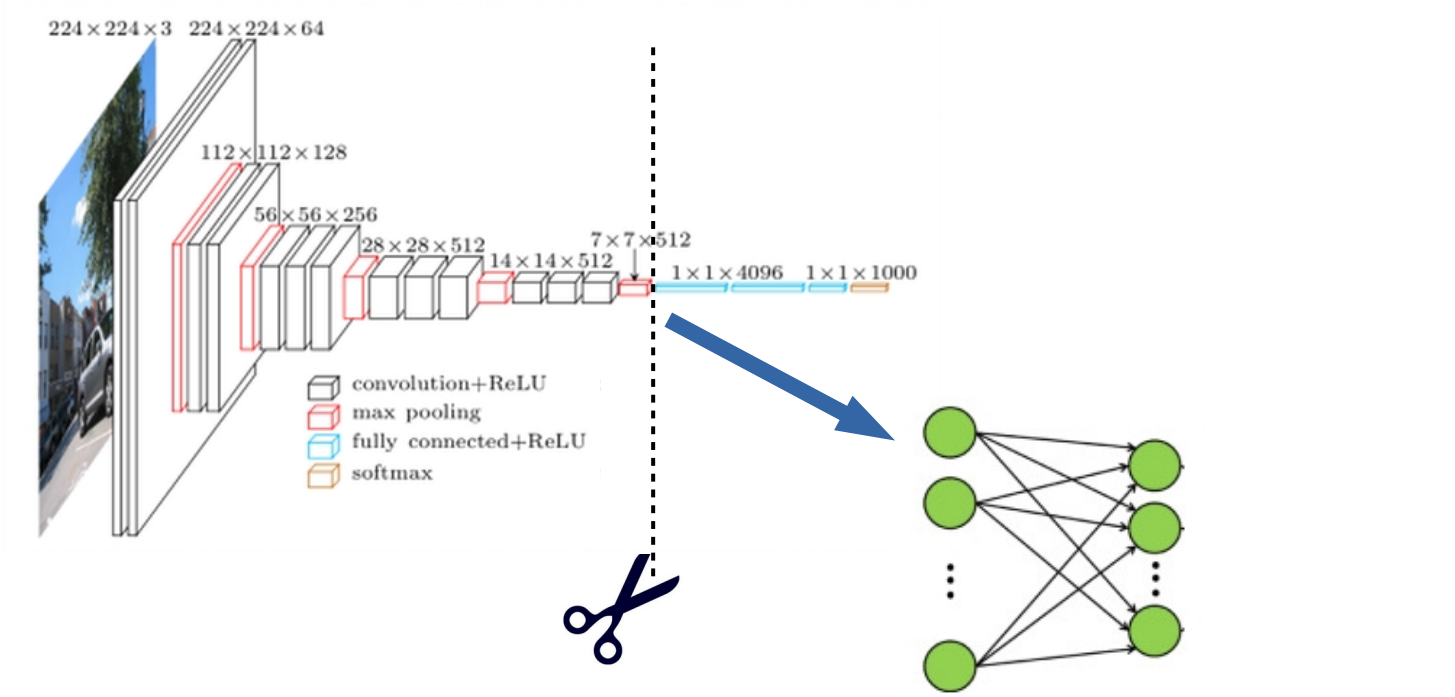
Démarche suivie : CNN + transfer learning

Utilisation de la technique de transfer learning : partir d'un réseau déjà pré-entraîné, et effectuer des ajustements pour l'adapter à notre problématique

⇒ Exemple ici en partant du réseau convolutif VGG16 :



Etape 1 : avec 20 classes, transfer learning de VGG16



Ré-entraînement de toutes les couches sauf les 5 premières

Suite de l’étape 1 ⇒ Pre processing

Resizing des images 224 x 224 (pour compatibilité avec réseau VGG16)

- Normalisation : soustraire la moyenne des valeurs de chaque canal R, G, B sur
- Imagenet

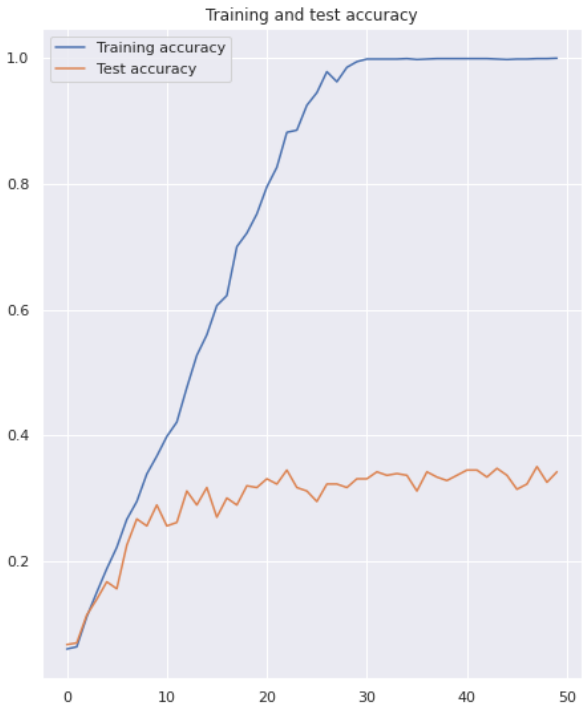
Suite de l’étape 1 ⇒ Hyper paramètres et métrique

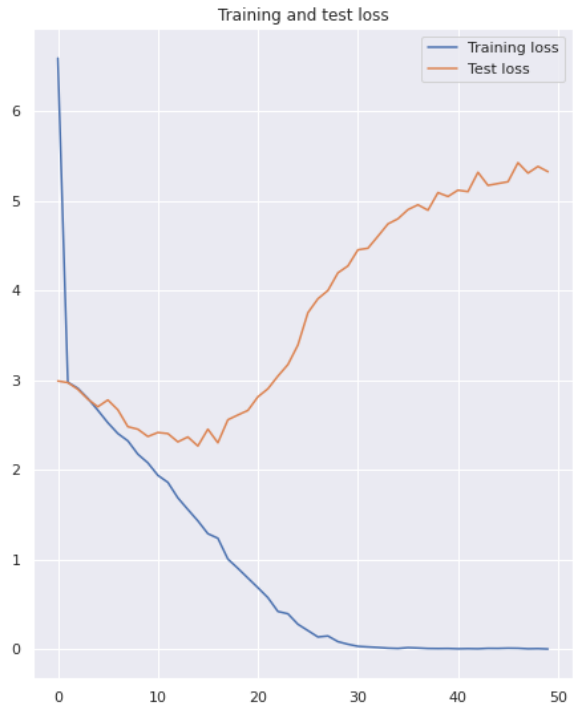
- Hyper paramètres :
  - Optimiser : Stochastic Gradient Descent
  - Learning rate : 0.0001
  - Momentum : 0.9
  - Fonction de coût : categorical cross entropy

- Métrique : accuracy

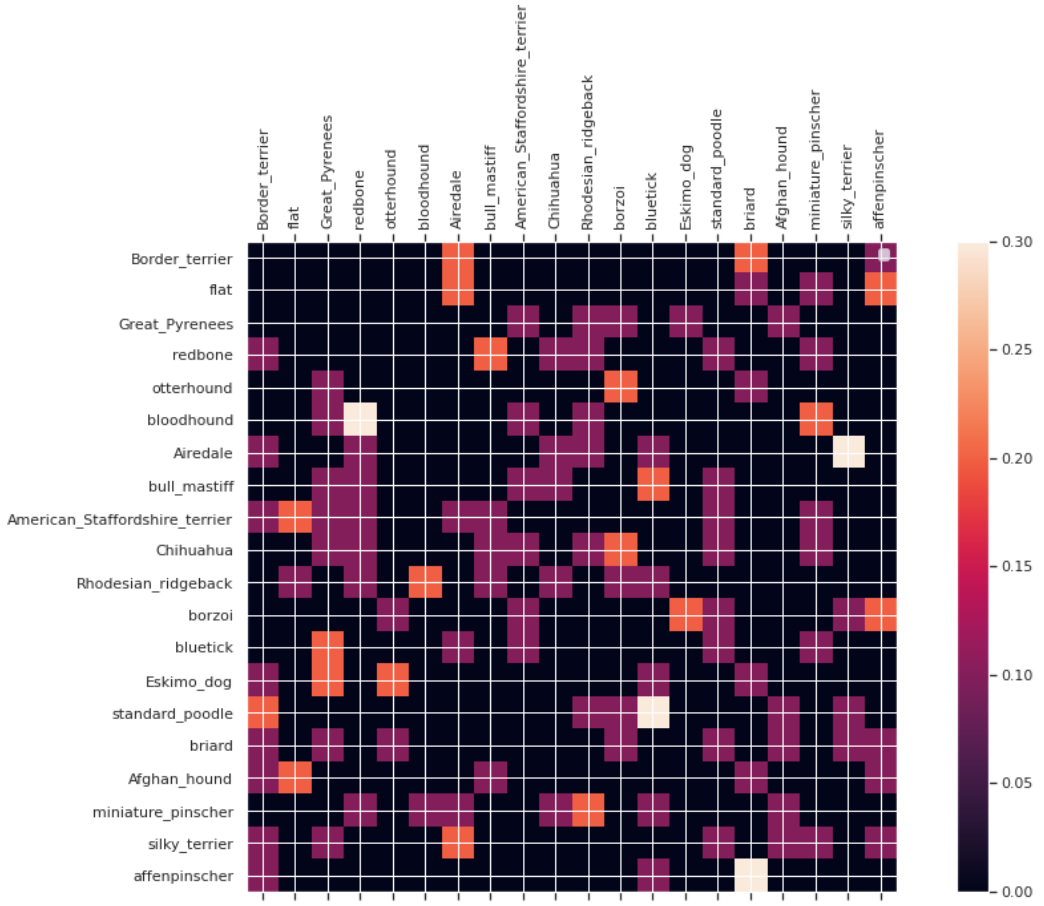
Suite de l’étape 1 ⇒ Résultats

- Accuracy sur test set : **0.29**





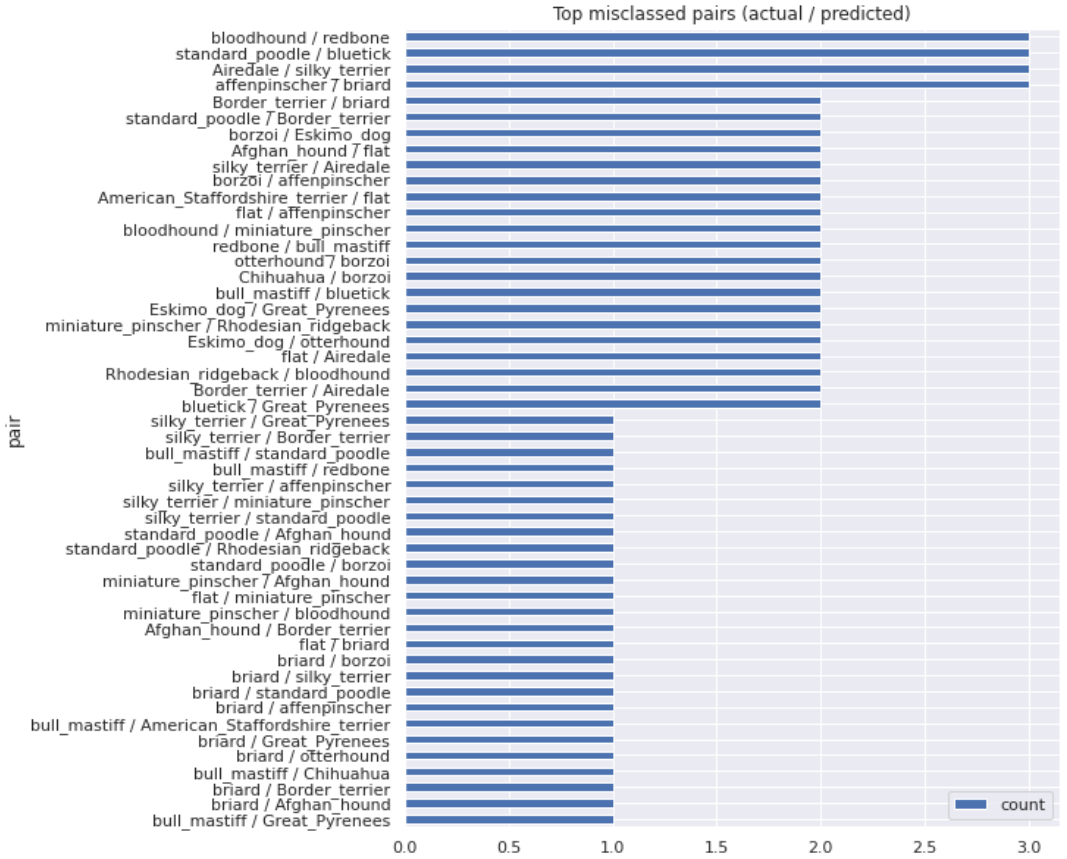
Suite de l’étape 1 ⇒ Résultats



Matrice de confusion

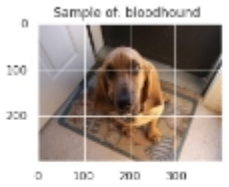
(La diagonale a été mise en noir pour mettre l’accent sur la visualisation des erreurs)

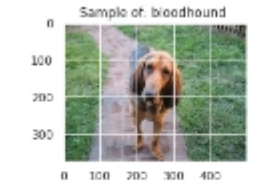
Suite de l’étape 1 ⇒ Résultats

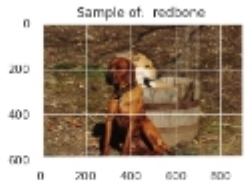


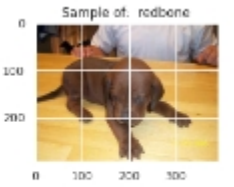
Top misclassed pairs

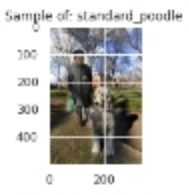
Suite de l’étape 1 ⇒ Exemples de classes confondues

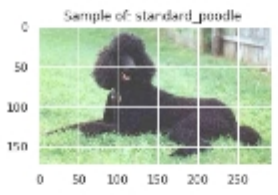


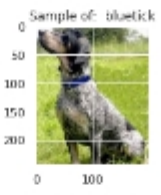


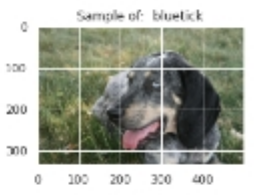


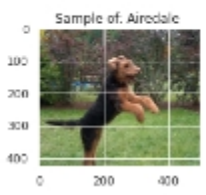


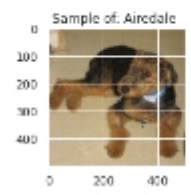


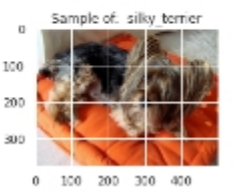




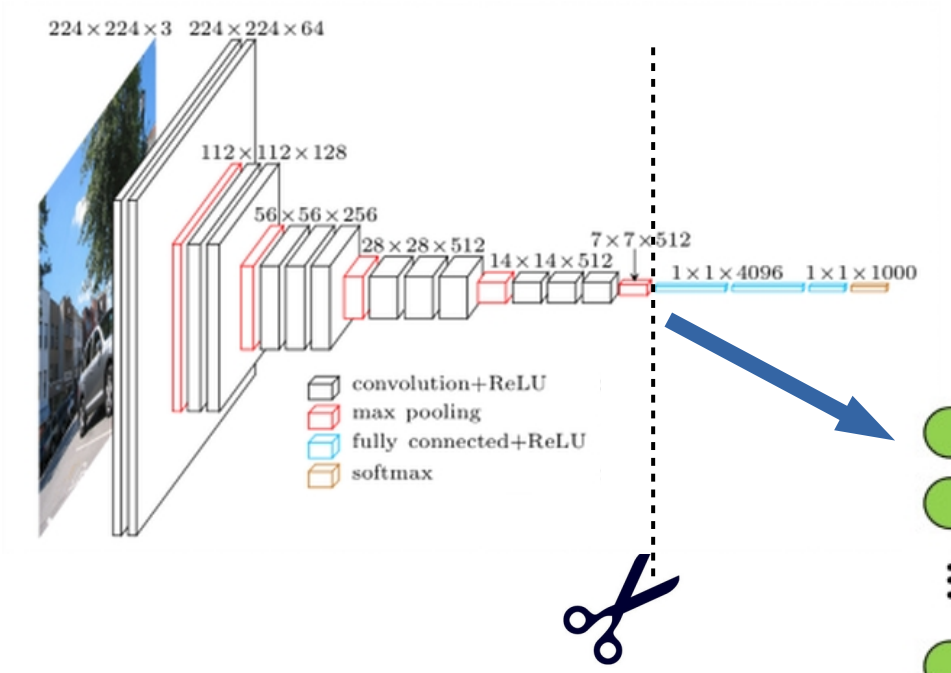


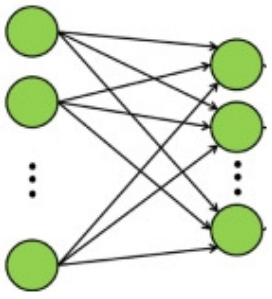






Etape 2 : 120 classes, transfer learning de VGG16





Flatten
25 088

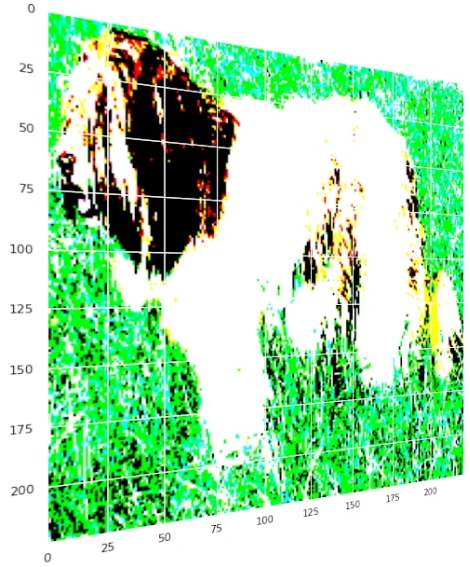
Dense (softmax output)
120

Ré-entraînement
Uniquement des 5 dernières couches
(incluant les 2 couches custom)

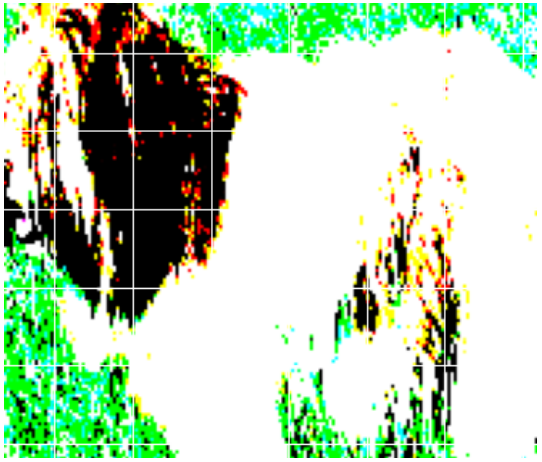
Suite de l’étape 2 ⇒ Pre processing

- Resizing des images 224 x 224 et normalisation comme pour l’étape 1
- Centrage des images (crop) sur les coordonnées de chaque chien
- Data augmentation :

Shear

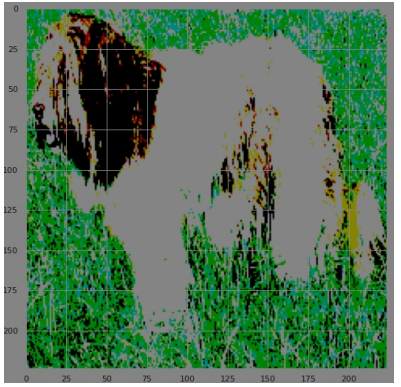


Zoom



Horizontal Flip

Brightness

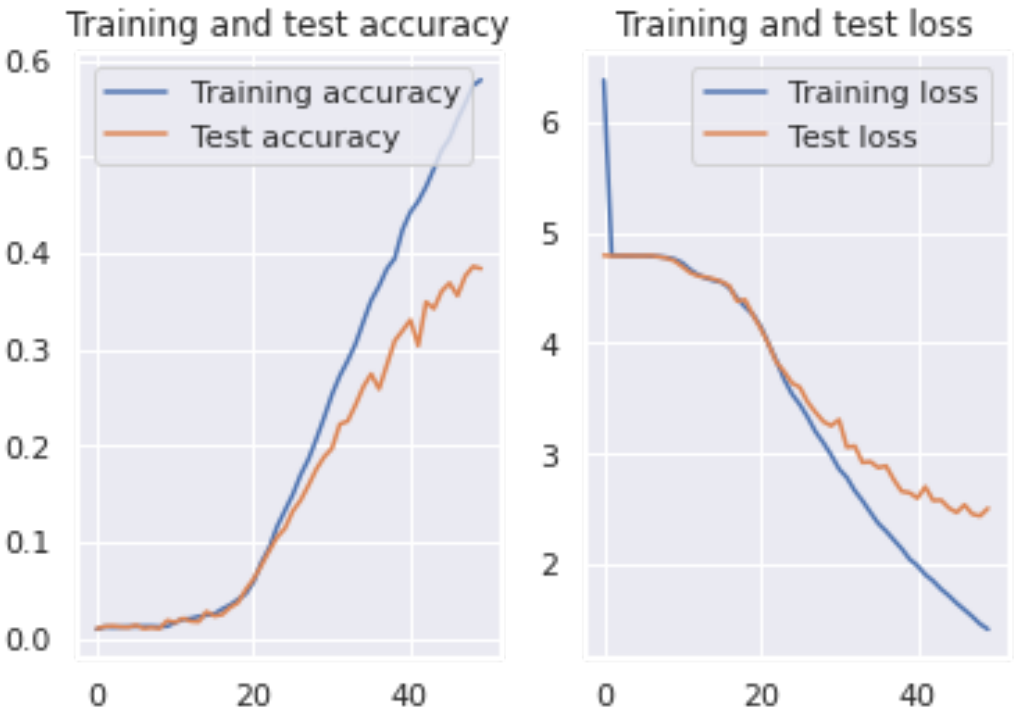


Suite de l’étape 2 $\Rightarrow$ Hyper paramètres et métrique

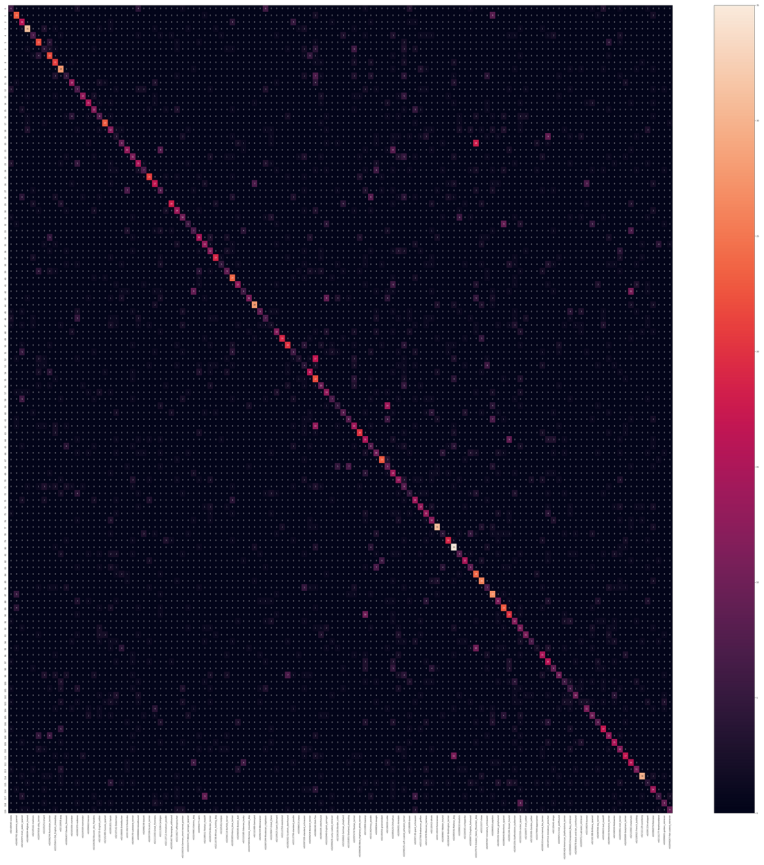
On a utilisé les mêmes hyper paramètres qu’à l’étape 1

Suite de l’étape 2 ⇒ Performance

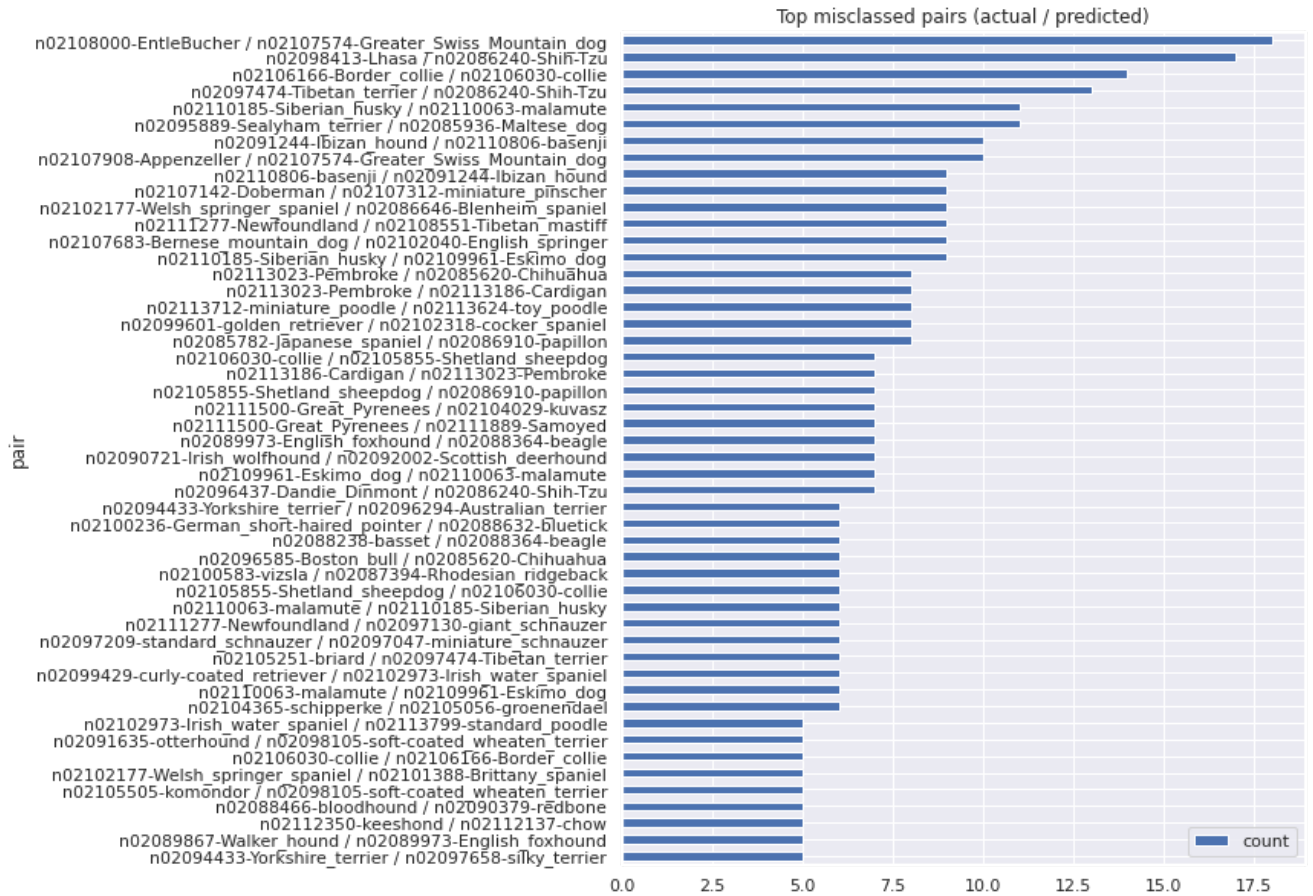
- Accuracy sur test set : **0.38**



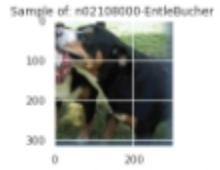
Suite de l’étape 2 ⇒ Matrice de confusion

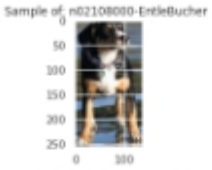


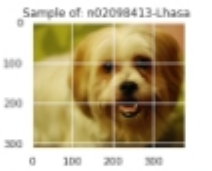
Suite de l’étape 2 ⇒ Top misclassed pairs

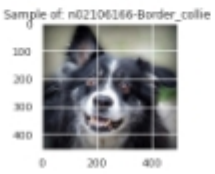


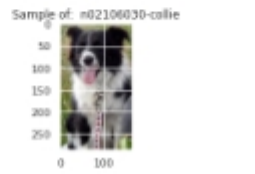
Suite de l’étape 2 ⇒ Exemples de top class mismatch

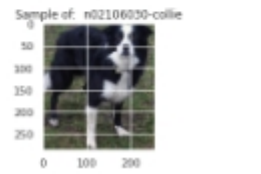




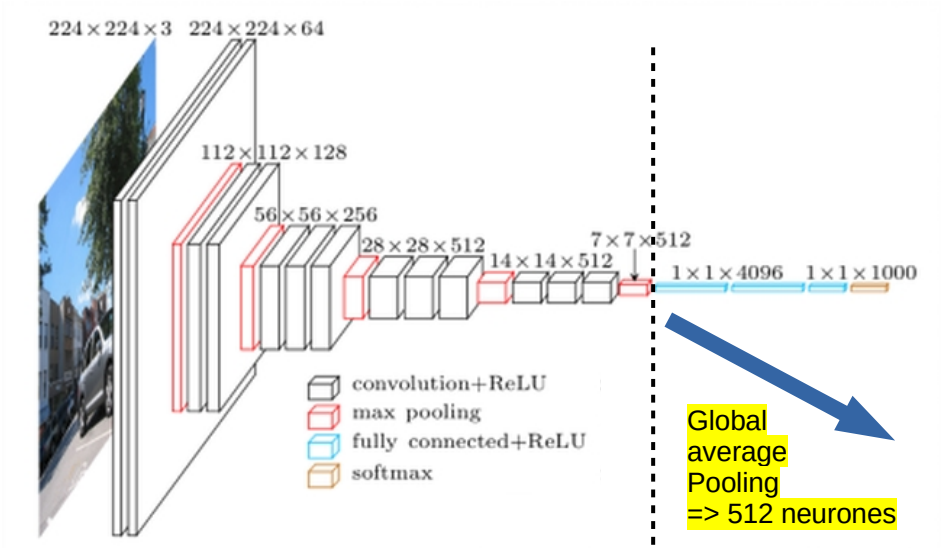








Etape 3 : avec des couches de classification + complexes



Global average Pooling => 512 neurones

Dense 1024

Dense 1024

Drop out 0.5

Dense 512

Dense (softmax output) 120

Pas de ré-entraînement du VGG16

Entrainement uniquement des nouvelles couches

Suite de l’étape 3 ⇒ Pre processing

- Resizing des images 224 x 224, normalisation, et data augmentation comme pour l’étape 2 (shear, zoom, horizontal flip, brightness)

Suite de l’étape 3 ⇒ Hyper paramètres

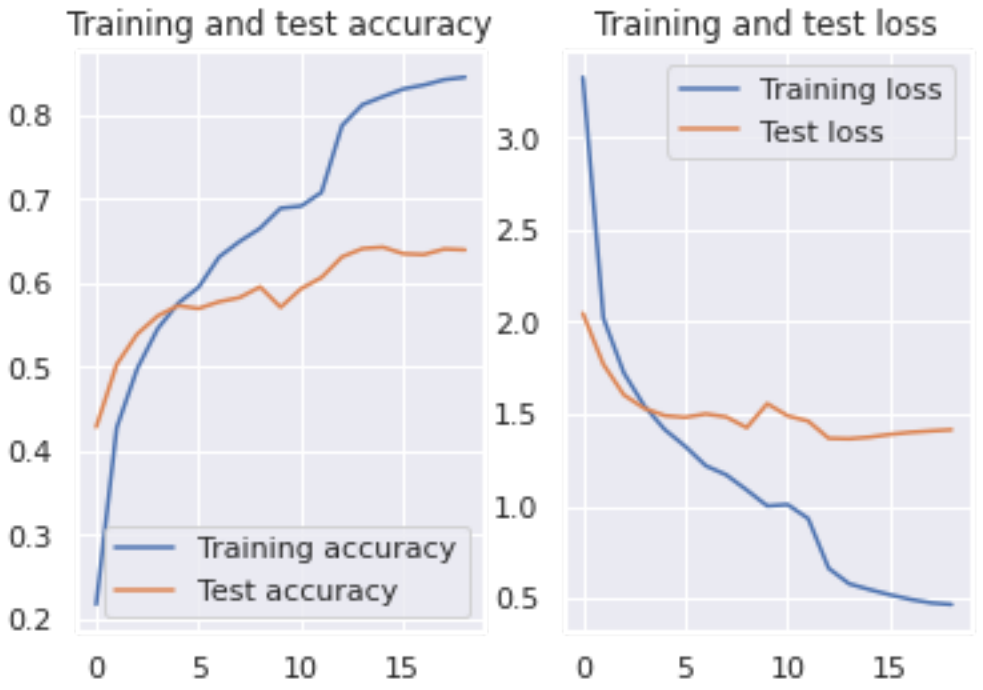
- Optimiser : Adam (= Adaptative Moment estimation)
  - Il s’agit d’une optimisation de la descente de gradient stochastique, qui conduit à une adaptation automatique de la learning rate en tenant compte de la moyenne des gradients des itérations précédentes

- Implémentation d’un early stopping (patience 5)

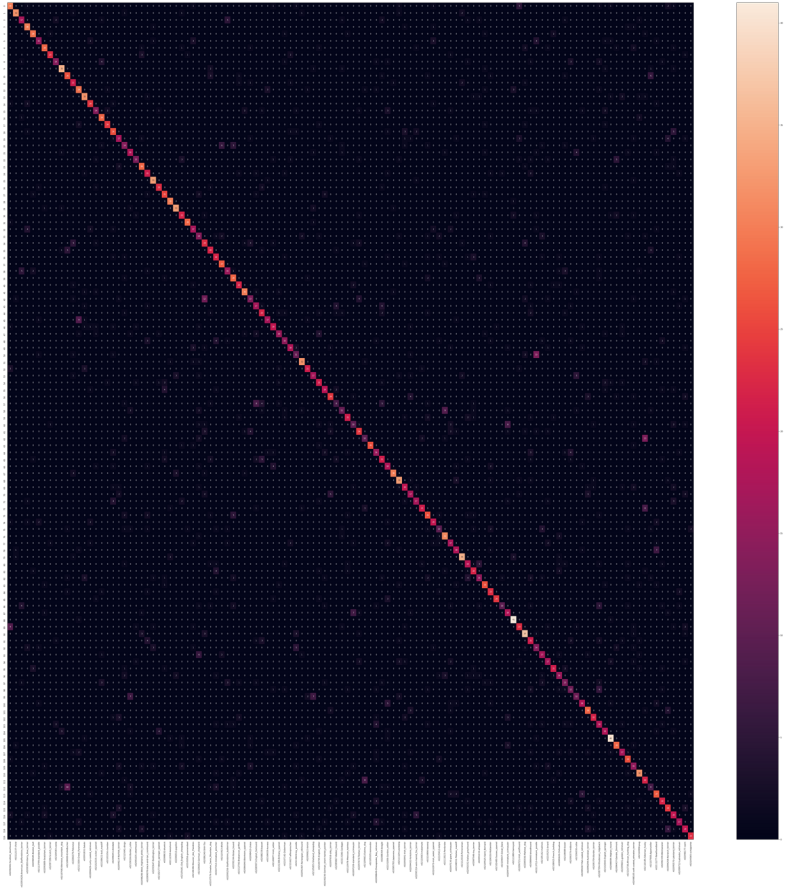
- Implémentation de Reduce LR On Plateau (factor 0.1, patience 3)

Suite de l’étape 3 ⇒ Performance

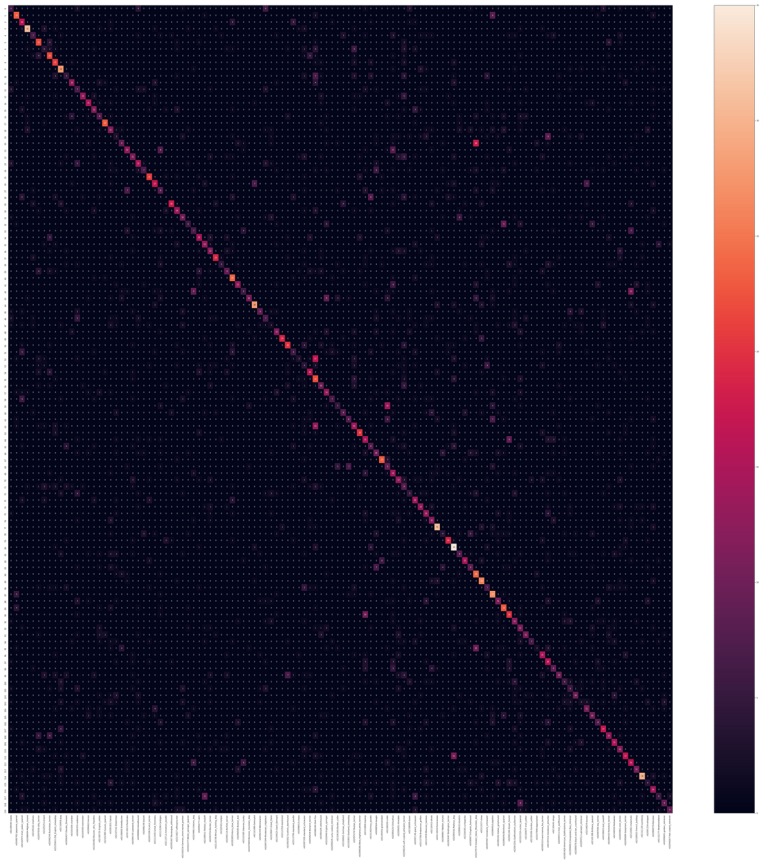
- Accuracy sur test set : **0.64**



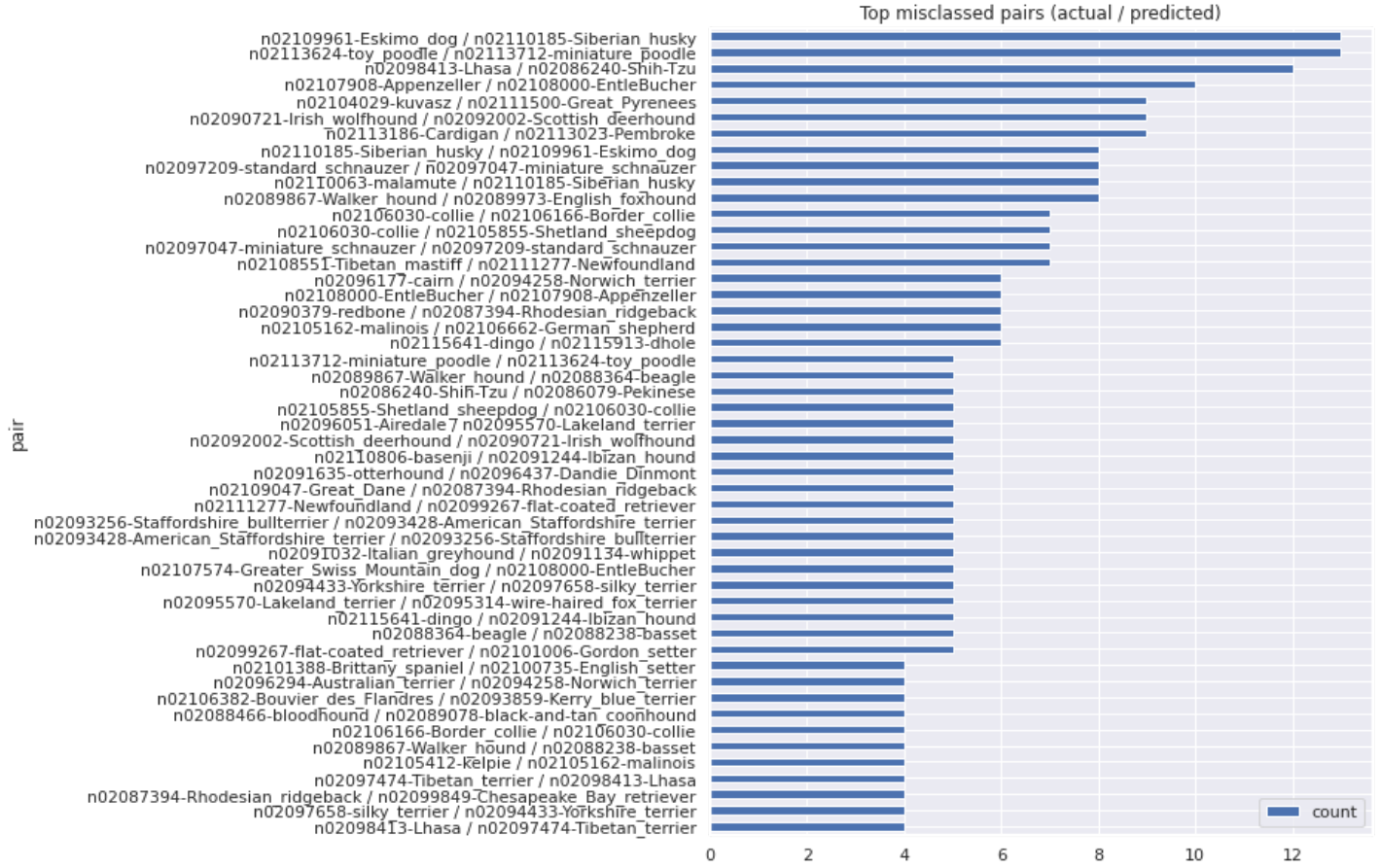
Suite de l’étape 3 ⇒ Matrice de confusion



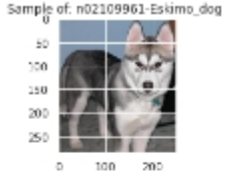
Suite de l’étape 3 ⇒ Matrice de confusion

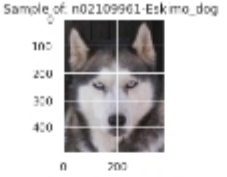


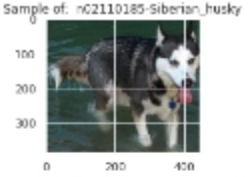
Suite de l’étape 3 ⇒ Top misclassed pairs

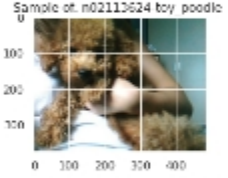


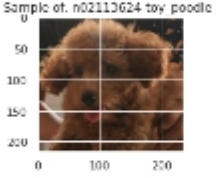
Suite de l’étape 3 ⇒ Exemples de top class mismatch

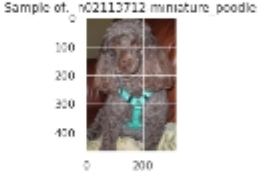


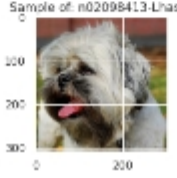


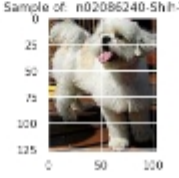


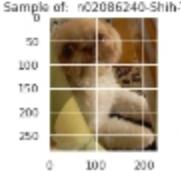




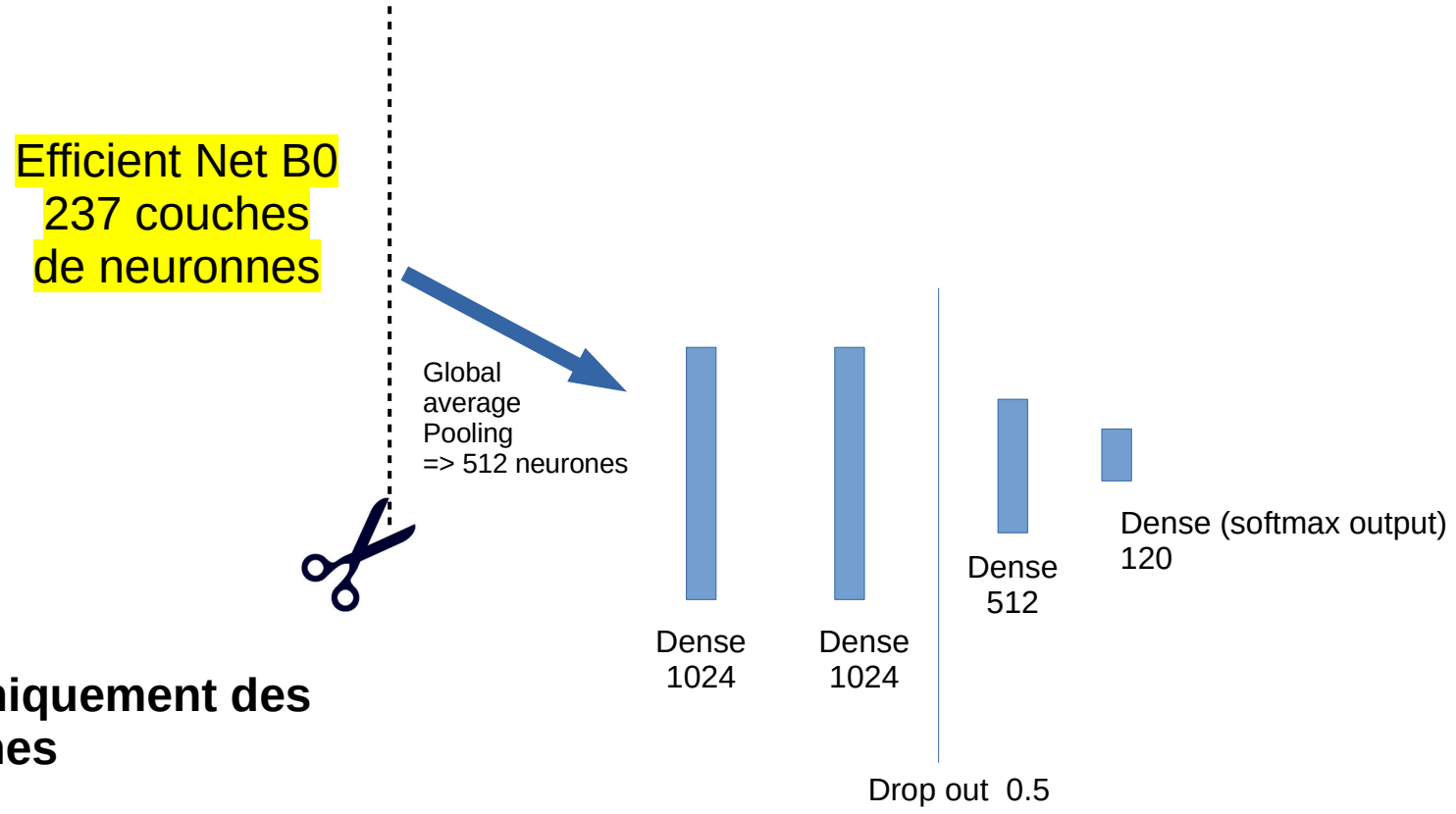








Etape 4 : utilisation d'Efficient Net B0 (modèle final)



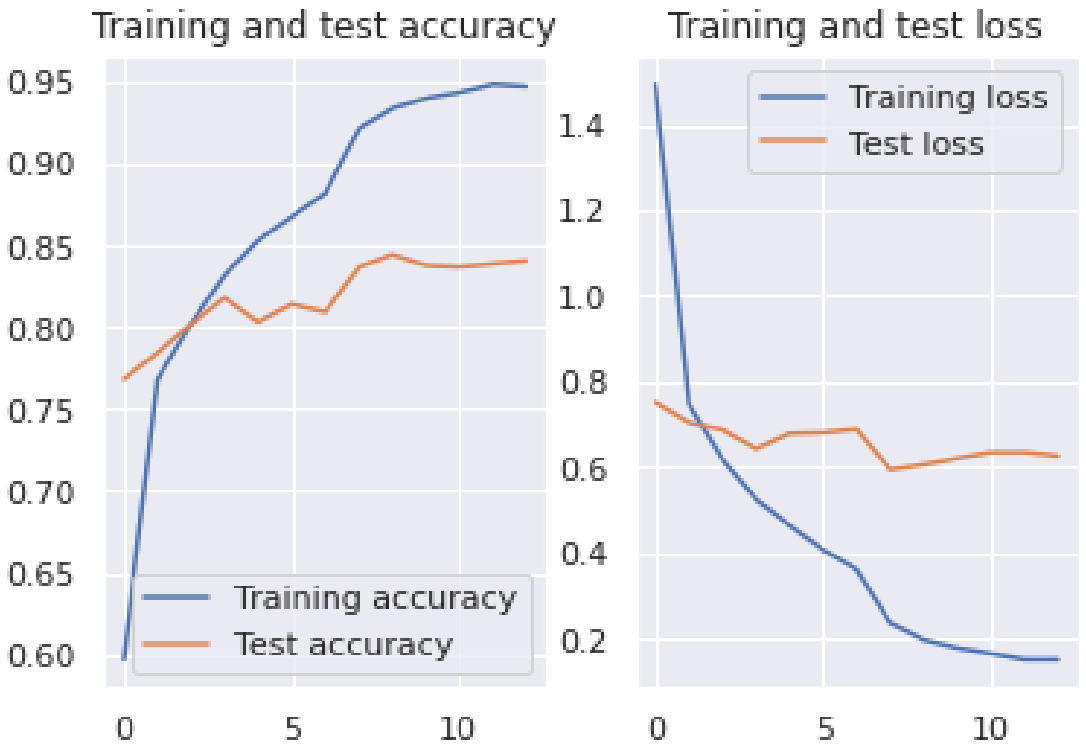
Entrainement uniquement des nouvelles couches

Suite de l’étape 4 ⇒ Pre processing, hyper paramètres

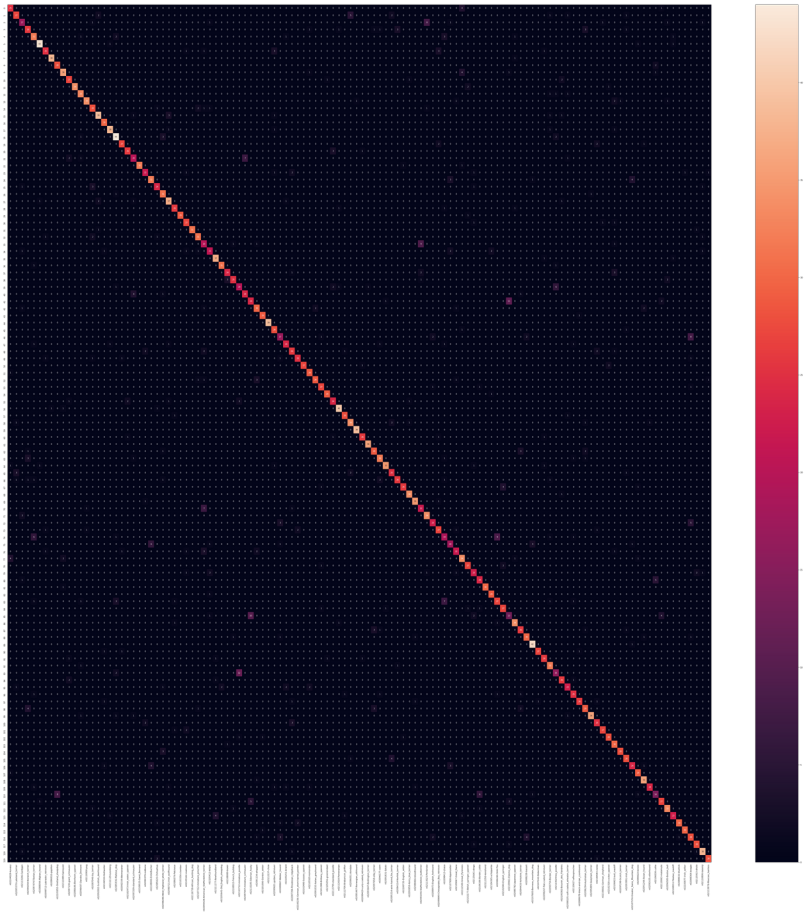
- Preprocessing et hyper paramètres inchangés par rapport à l’étape précédente

Suite de l’étape 4 ⇒ Performance

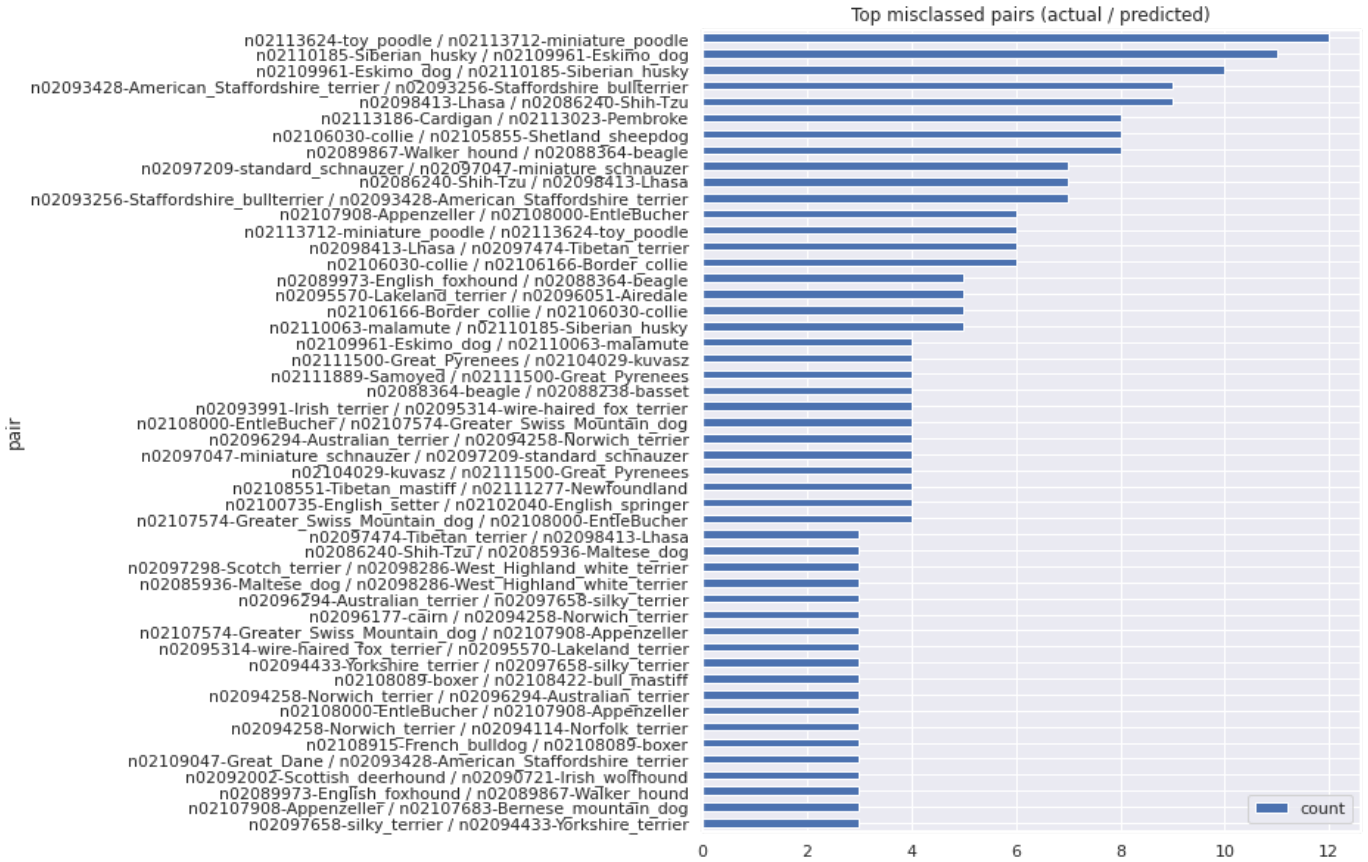
- Accuracy sur test set : **0.84**



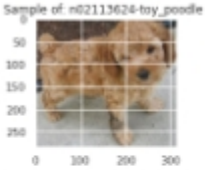
Suite de l’étape 4 ⇒ Matrice de confusion

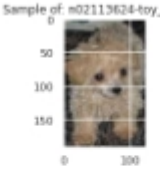


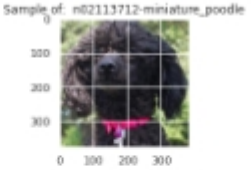
Suite de l’étape 4 ⇒ Top misclassed pairs

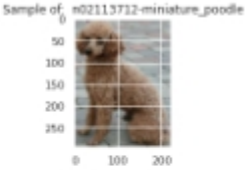


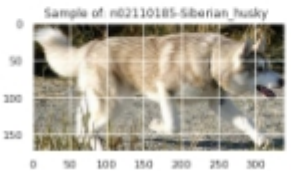
Suite de l’étape 4 ⇒ Exemples de top class mismatch

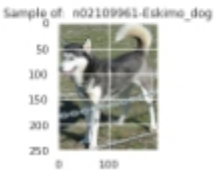


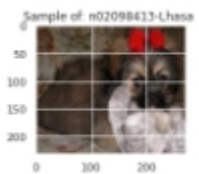


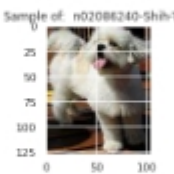


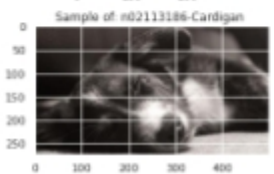


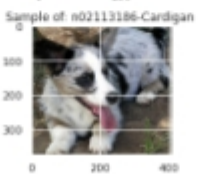


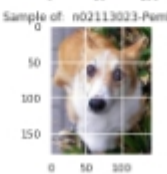


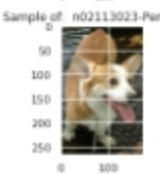






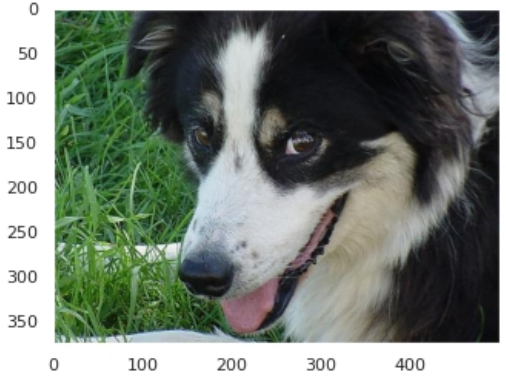






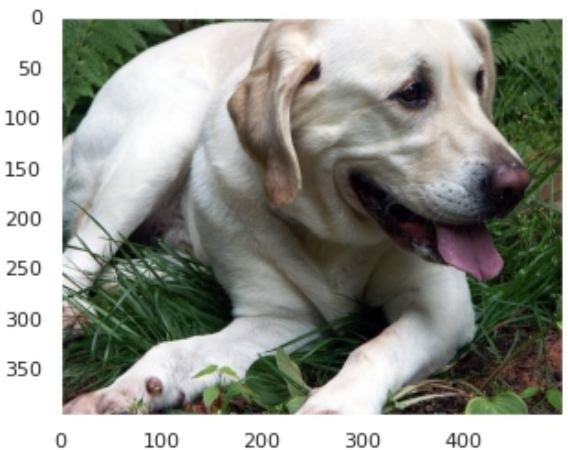
Suite de l’étape 4 ⇒ Exemples de prédictions

```
../input/stanford-dogs-dataset/images/Images/n02106166-Border_collie/n02106166_1032.jpg
0.9183174   :   (81, 'n02106166-Border_collie')
0.06897815   :   (80, 'n02106030-collie')
0.004547952   :   (79, 'n02105855-Shetland_sheepdog')
0.0033916084   :   (112, 'n02113186-Cardigan')
0.0010028793   :   (89, 'n02107908-Appenzeller')
```



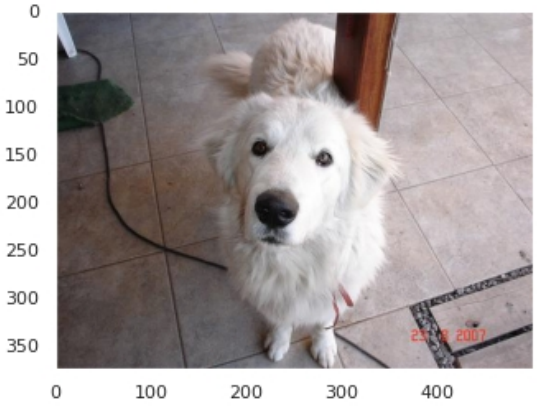
Suite de l’étape 4 ⇒ Exemples de prédictions

```
../input/stanford-dogs-dataset/images/Images/n02099712-Labrador_retriever/n02099712_1383.jpg
0.99995613   :   (57, 'n02099712-Labrador_retriever')
2.6873056e-05   :   (11, 'n02088364-beagle')
5.9290846e-06   :   (56, 'n02099601-golden_retriever')
3.7967495e-06   :   (25, 'n02091831-Saluki')
1.3143558e-06   :   (54, 'n02099267-flat-coated_retriever')
```



Suite de l’étape 4 ⇒ Exemples de prédictions

```
../input/stanford-dogs-dataset/images/Images/n02104029-kuvasz/n02104029_1206.jpg
0.7699173   :   (105, 'n02111500-Great_Pyrenees')
0.1730538   :   (18, 'n02090622-borzoi')
0.035787273   :   (71, 'n02104029-kuvasz')
0.007606616   :   (80, 'n02106030-collie')
0.0055934787   :   (106, 'n02111889-Samoyed')
```



Suite de l’étape 4 ⇒ Exemples de prédictions

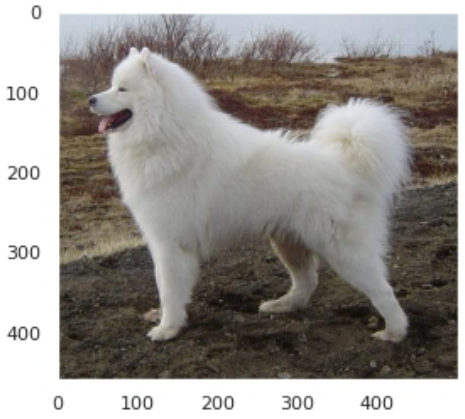
```
../input/stanford-dogs-dataset/images/Images/n02097658-silky_terrier/n02097658_10020.jpg
0.9823512    :   (50, 'n02097658-silky_terrier')
0.011406802    :   (42, 'n02096294-Australian_terrier')
0.0062335595    :   (36, 'n02094433-Yorkshire_terrier')
2.0439363e-06    :   (41, 'n02096177-cairn')
1.6896288e-06    :   (49, 'n02097474-Tibetan_terrier')
```

Suite de l’étape 4 ⇒ Exemples de prédictions

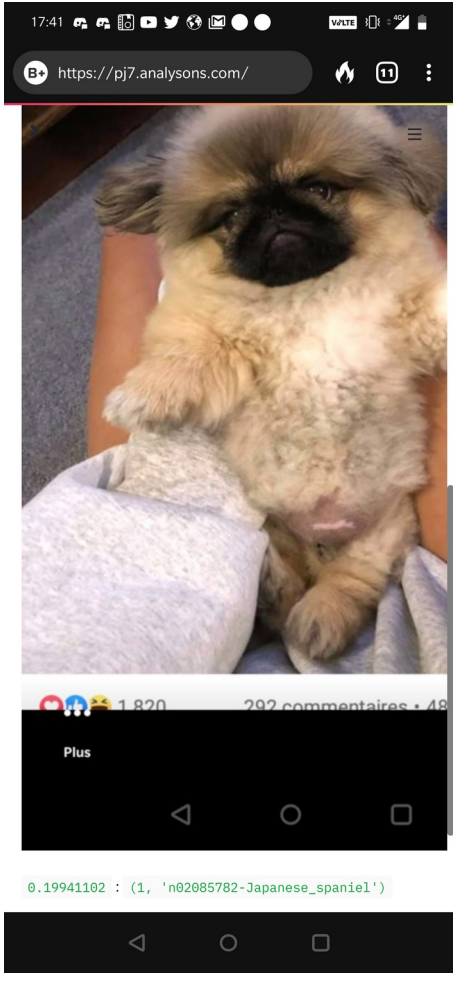
```
../input/stanford-dogs-dataset/images/Images/n02106550-Rottweiler/n02106550_10222.jpg
0.99943286   :   (83, 'n02106550-Rottweiler')
0.00036354657   :   (86, 'n02107312-miniature_pinscher')
0.0001462628   :   (85, 'n02107142-Doberman')
3.64507e-05   :   (14, 'n02089078-black-and-tan_coonhound')
5.6855583e-06   :   (87, 'n02107574-Greater_Swiss_Mountain_dog')
```

Suite de l’étape 4 ⇒ Exemples de prédictions

```
../input/stanford-dogs-dataset/images/Images/n02111889-Samoyed/n02111889_1363.jpg
0.9999939   :   (106, 'n02111889-Samoyed')
4.231638e-06   :   (105, 'n02111500-Great_Pyrenees')
8.365155e-07   :   (97, 'n02109961-Eskimo_dog')
1.9720757e-07   :   (80, 'n02106030-collie')
1.6420286e-07   :   (99, 'n02110185-Siberian_husky')
```



Suite de l’étape 4 ⇒ Exemples de prédictions



Ce pékinois est reconnu correctement à droite (cadrage précis sur le chien)

Il est moins bien reconnu à gauche (cadrage moins précis)
=> confusion avec le japanese spaniel qui est une race proche

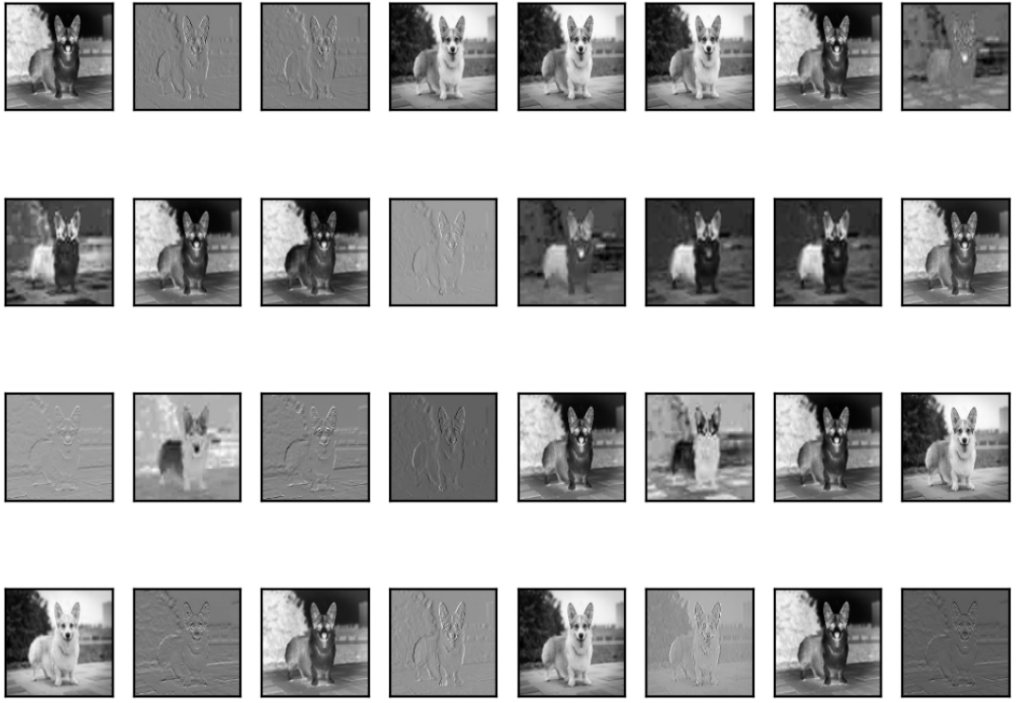
Interface utilisateur pour le modèle final

https://pj7.analysons.com/

Interface utilisateur pour le modèle final

https://pj7.analysons.com/

```
0.9310391 : (111, 'n02113023-Pembroke')
0.06803509 : (112, 'n02113186-Cardigan')
0.0007404797 : (0, 'n02085620-Chihuahua')
9.819656e-05 : (6, 'n02086910-papillon')
1.5682062e-05 : (7, 'n02087046-toy_terrier')
```



Perspectives pour améliorer le modèle

- Gérer le problème de l'environnement (le fond des images) : personnes, gazon, …
  - Utiliser les « saliency maps » afin d'identifier la localisation des chiens
  - Travailler sur le dataset brut, sans le recentrage (crop)

- Gérer le problème lié à certaines fortes variation intra classes vs faible variation inter classe

  ⇒ Effectuer du boosting sur les paires de classes les plus confondues

  ⇒ Par exemple : utilisation des Fully Convolutional Attention Networks (FCANs)

  https://arxiv.org/pdf/1603.06765.pdf

- Corriger les erreurs d'étiquettage dans le dataset


QUESTION
ANSWER