# Classification d’images

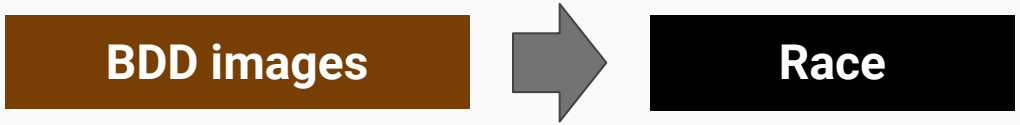
Xavier Barbier - Février 2021

# Sommaire

1. Contexte
2. Projet
3. Données
4. CNN
5. Transfert learning
6. API
7. Conclusion

# Contexte et projet :

- association de protection des animaux.
- temps de référencer les images des animaux qu'ils ont accumulées depuis plusieurs années.




1. **Prétraitement**
2. **CNN**
3. **Transfert learning**
4. **Point d'entrée API (web app)**

**Python / Notebook Jupyter / Colab / CloudReady**

# DONNÉES

# DONNÉES

**Stanford Dogs Dataset**

L'ensemble de données de Stanford Dogs contient des **20,580 images de 120 races de chiens** du monde entier.

Cet ensemble de données a été construit à l'aide d'images et d'annotations d'ImageNet pour la tâche de catégorisation fine des images.

cairn

Tibetan_mastiff

Greater_Swiss_Mountain_dog

redbone

English_setter

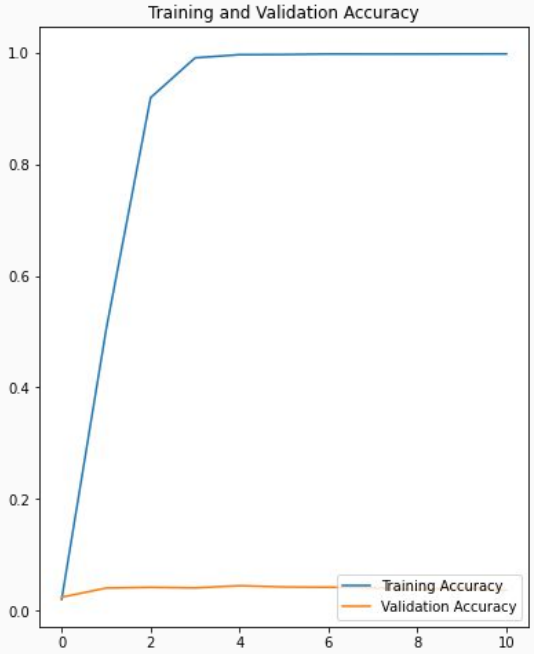
# CNN

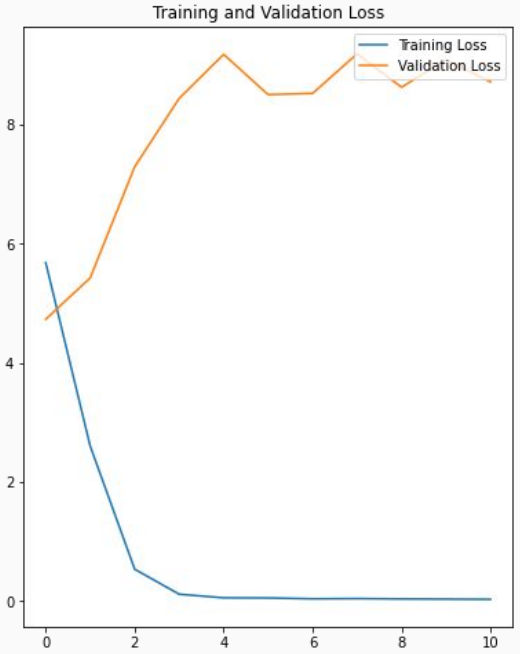
# Démarche

- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

Conv2D - 16 (3x3)

MaxPooling (2x2)



# Démarche

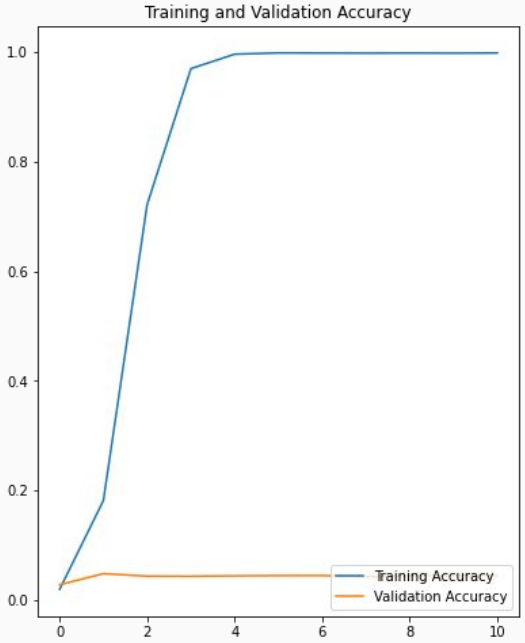
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

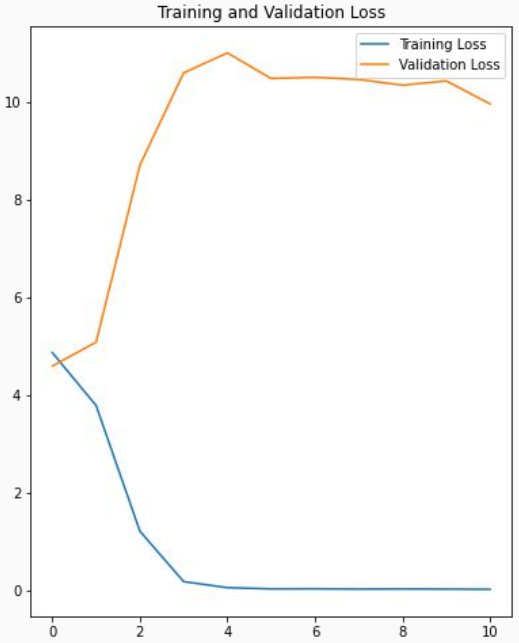
Conv2D - 16 (3x3)

MaxPooling (2x2)

Conv2D - 32 (3x3)

MaxPooling (2x2)



# Sur-apprentissage

- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

**Image augmentation**

**Dropout**

# Image augmentation

**Rescaling** | **Translation** | **Rotation** | **Zoom** | **Shear**

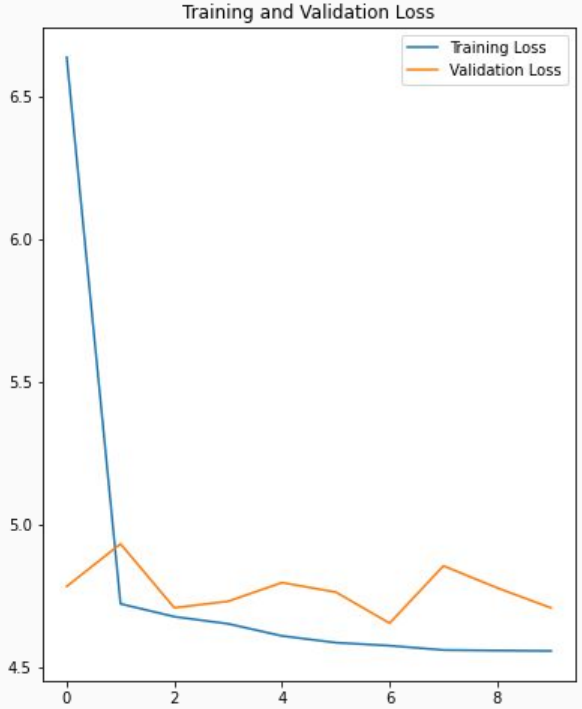
# Sur-apprentissage

Conv2D - 16 (3x3)

MaxPooling (2x2)

Image augmentation

Dropout

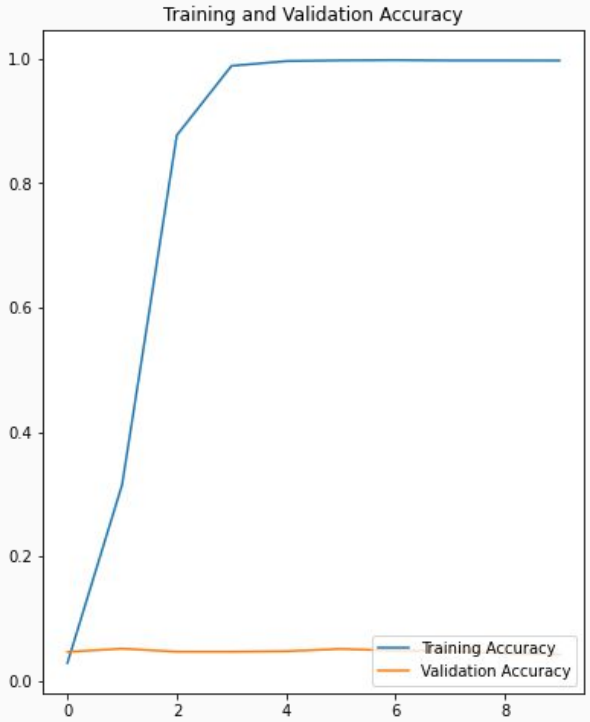
# Sur-apprentissage

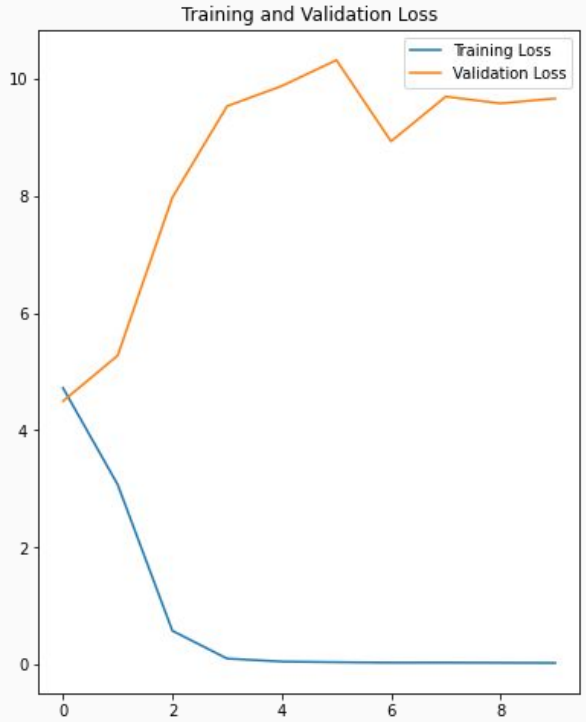
Conv2D - 16 (3x3)

MaxPooling (2x2)

Image augmentation

Dropout



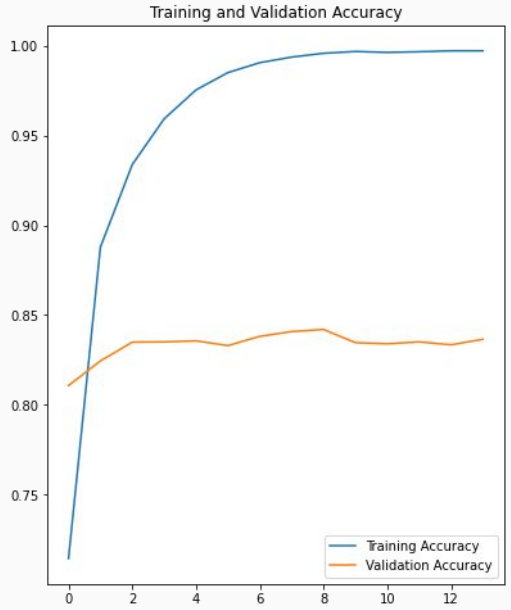
# Transfert learning

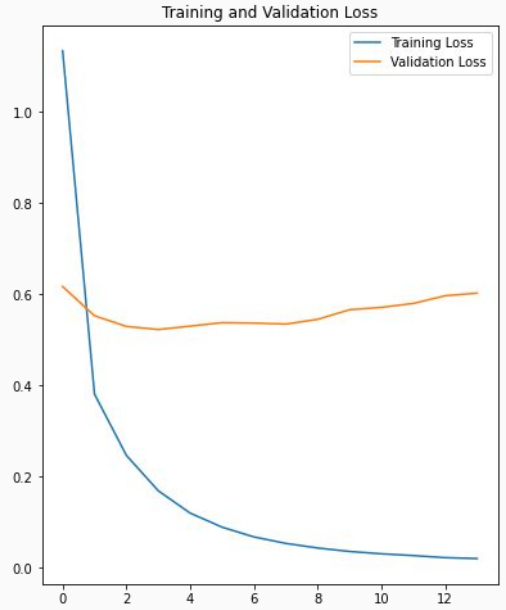
# Transfert learning

TensorFlow Hub

- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
keras_layer (KerasLayer)     (None, 1280)              2257984
_________________________________________________________________
dense (Dense)                (None, 120)               153720
=================================================================
Total params: 2,411,704
Trainable params: 153,720
Non-trainable params: 2,257,984
_________________________________________________________________
```



# Transfert learning


- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

Dense (120, relu)

Image augmentation

Dropout

Normalisation

# Transfert learning

TensorFlow Hub

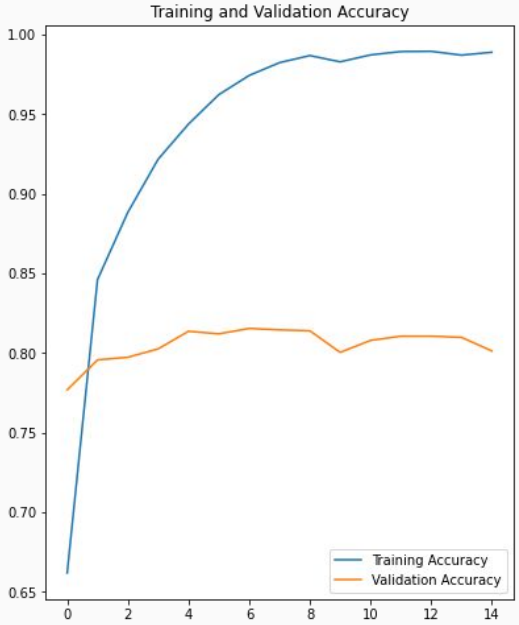
- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

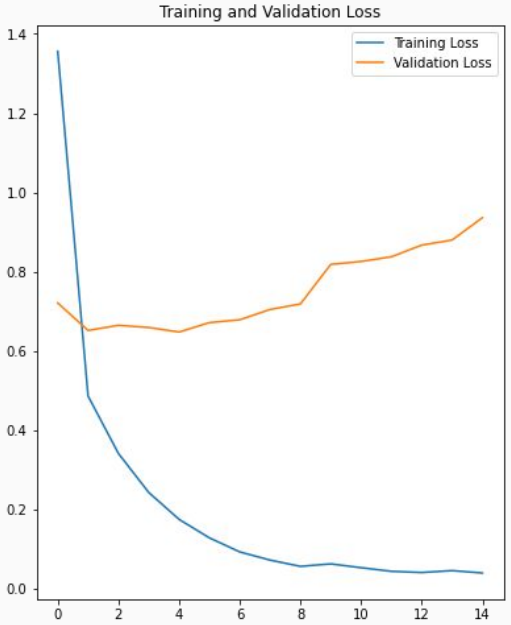
Dense (120, relu)

Image augmentation

Dropout

Normalisation



# Transfert learning

TensorFlow Hub

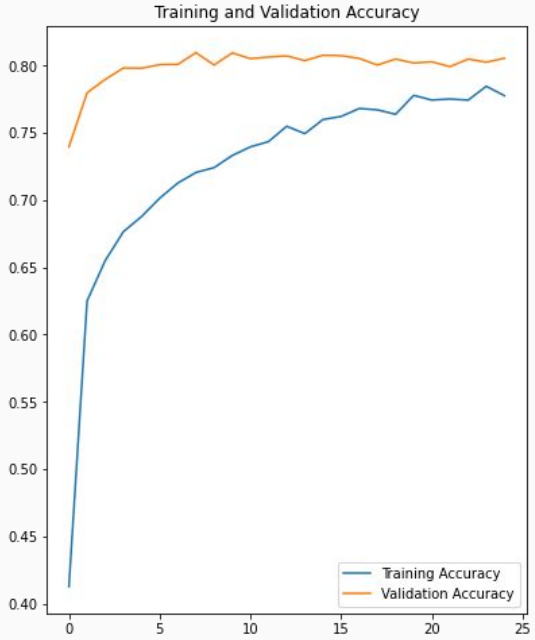
- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

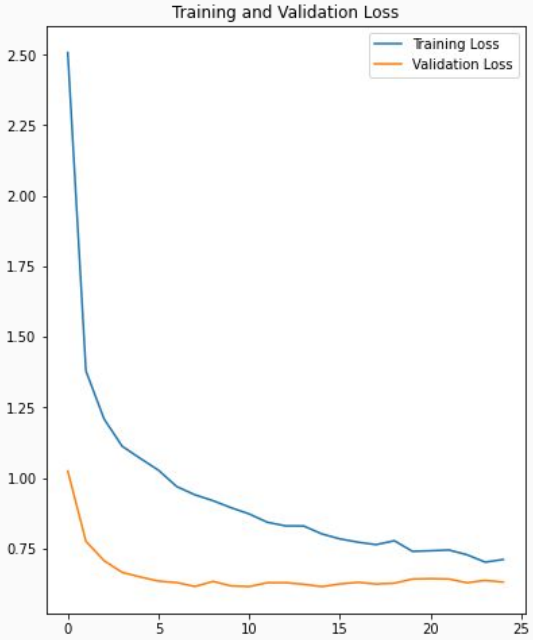
Dense (120, relu)

Image augmentation

Dropout

Normalisation



# Transfert learning

TensorFlow Hub

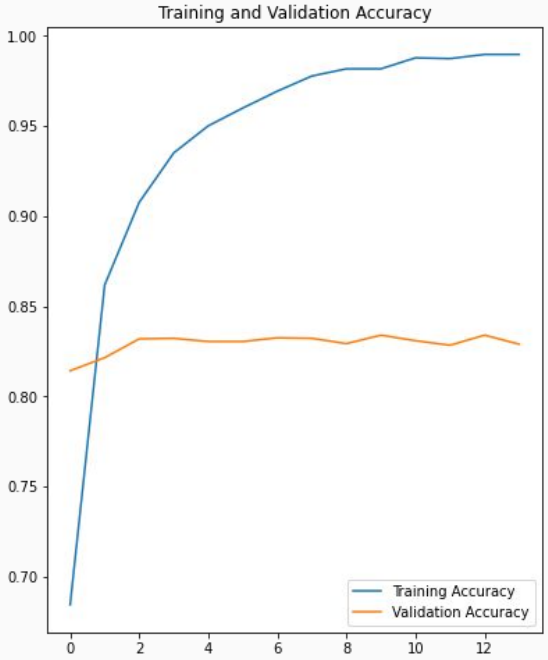
- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

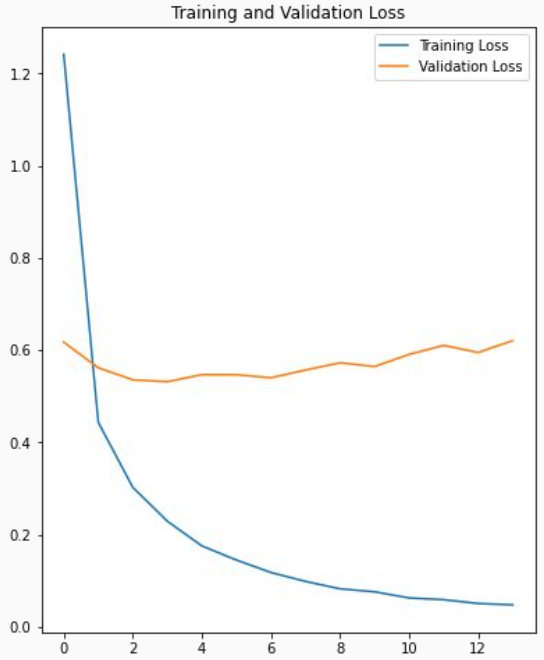
Dense (120, relu)

Image augmentation

Dropout

Normalisation



# Transfert learning

TensorFlow Hub

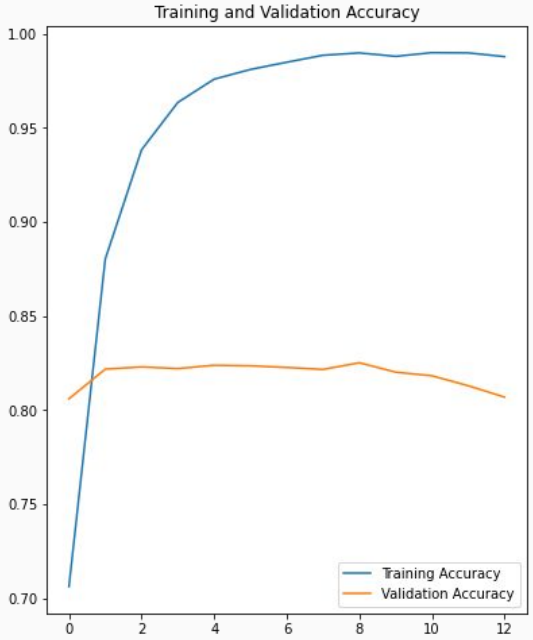
- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

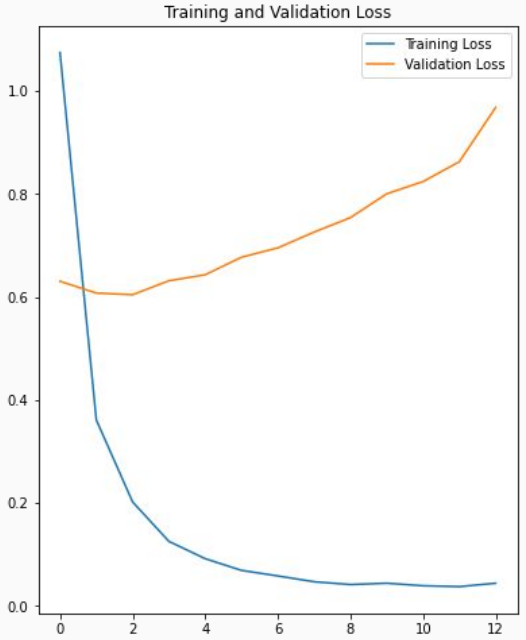
Dense (120, relu)

Image augmentation

Dropout

Normalisation

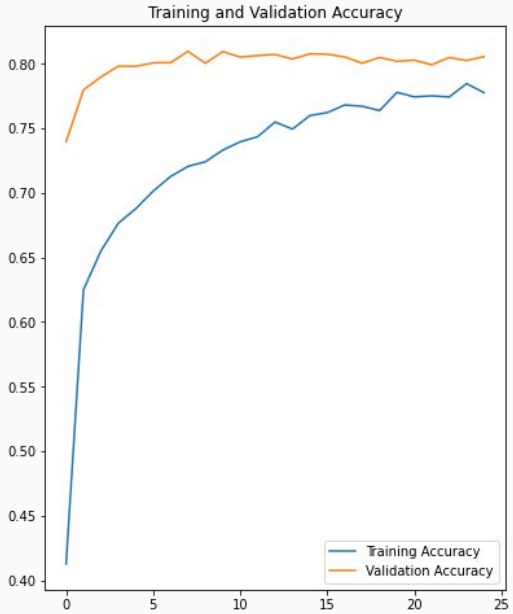


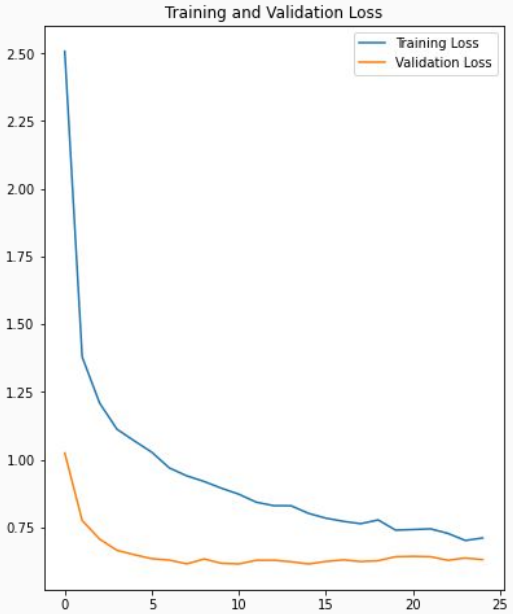



# Transfert learning

- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

**Image augmentation**



# Hypertuning

- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

**Image augmentation**

**Keras tuner**

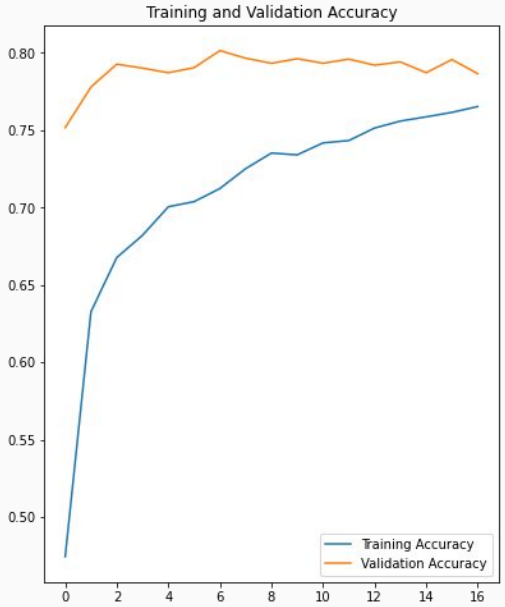
**lr = [0.1, 0.01,0.001]**

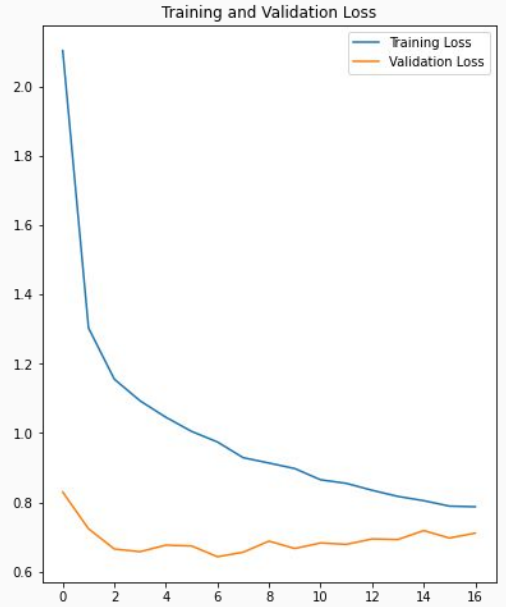
# Hypertuning

- MobilNet V2
- loss = Categorical_Crossentropy
- Callback “val_loss”, patience = 10
- Validation split = 0.33
- Dense layer (120, softmax)

**Image augmentation**

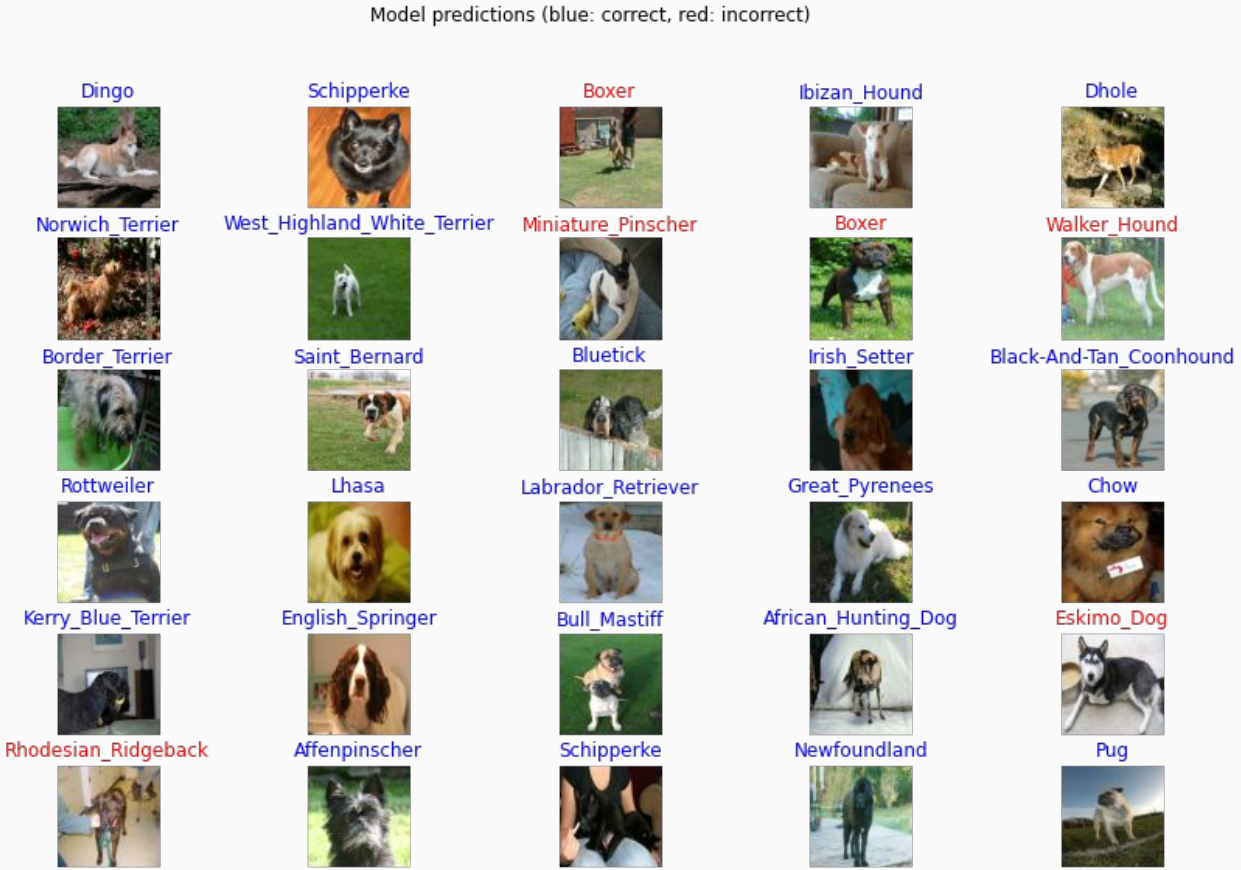
**lr = 0.001**




**Accuracy = 0.80**

# Prédictions



# API

# API

Dash

**MobilNet V2**

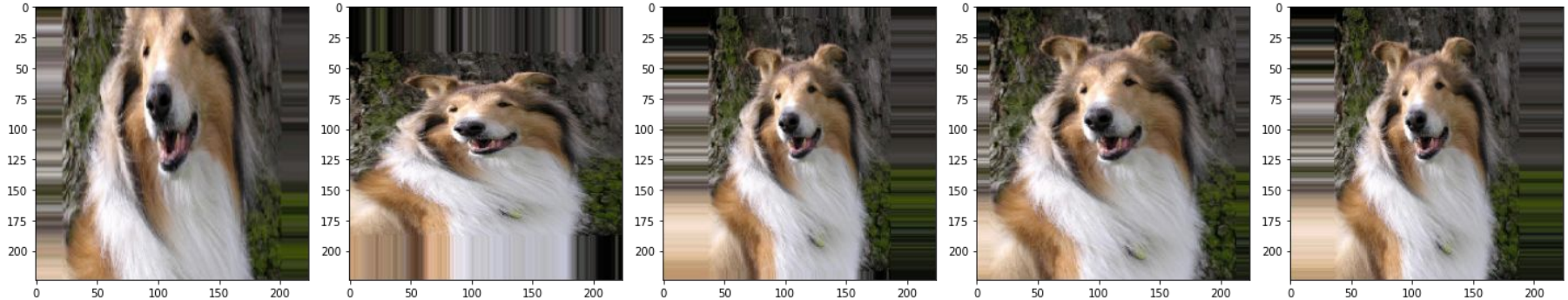
GitHub → heroku

https://dogs-prediction.herokuapp.com/

# CONCLUSION

TensorFlow Hub

**Accuracy = 0.80**



Dash

heroku

# PERSPECTIVES

# Perspectives

- Autres modèles de transfert learning
- Whitening
- Image augmentation plus faible
- Différentes librairies (ex = PyTorch)

# MERCI !

@xavbarbier

https://www.linkedin.com/in/barbierxavier/

https://github.com/xavierbarbier/

contact@xavierbarbier.com