

# Projet 6 : Soutenance

Classez des images à l'aide d'algorithmes de Deep Learning

Gaëtan PELLETIER

# Sommaire

- Problématique, interprétation et pistes de recherches
- Nettoyage et exploration des données, feature engineering
- Création d'un CNN
- Utilisation du *Transfer Learning*
- Modèle final
- Déploiement d'une application
- Synthèse

# Projet 6 : Soutenance

Problématique,  
interprétation  
et pistes de recherches

# Problématique

D'après les images provenant de *Stanford Dogs Dataset*, la problématique est :

- Comment pouvons-nous prédire la race d'un chien présent sur une photo ?

# Interprétation

Comment pouvons-nous prédire la race d'un chien présent sur une photo ?

- Analyse de la **qualité** des images
- **Transformations** des images
- Mise en place d'**algorithmes de prédictions**
- **Évaluation** de la qualité des prédictions

# Pistes de recherche envisagées

- **Analyse** de la qualité des images
- **Data augmentation**
- Modélisation d'un **CNN**
- Modélisation par *transfer learning*
- Choix du **meilleur** modèle
- Déploiement d'une **application**

# Projet 6 : Soutenance

Nettoyage et exploration  
des données,  
feature engineering

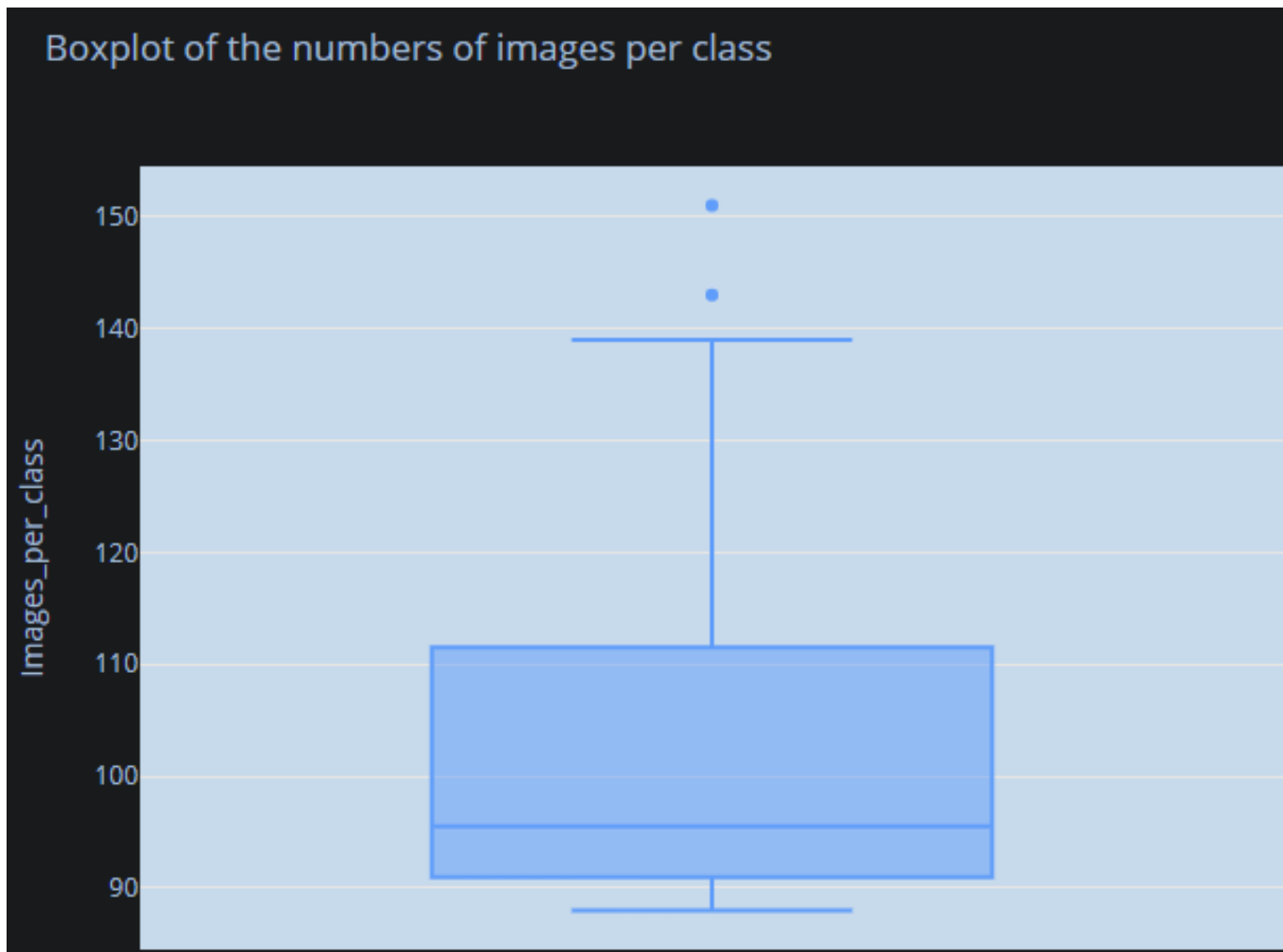
# Nettoyage des données

- **Importation** du jeu d'images *Stanford Dogs Dataset*
- Vérification la **qualité** des images :
  - Le jeu de données contient bien 120 classes
  - Chaque classe **ne possède pas** de photos entièrement **noires ou blanches**
- Les « problèmes » de contraste, luminosité, etc n'impactent pas la qualité du jeu d'images



# Exploration des données

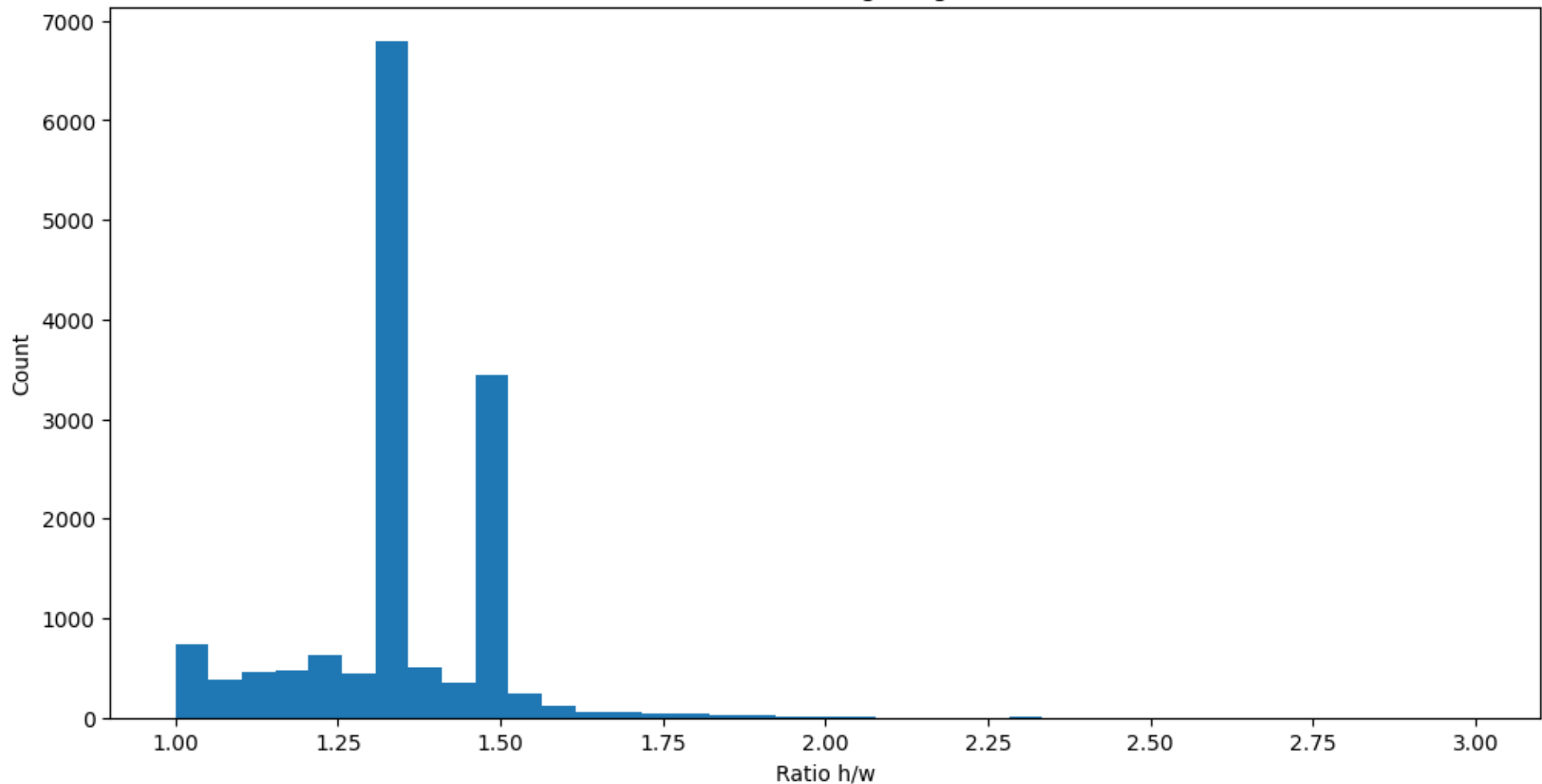
## Nombre d'images par classe



# Exploration des données

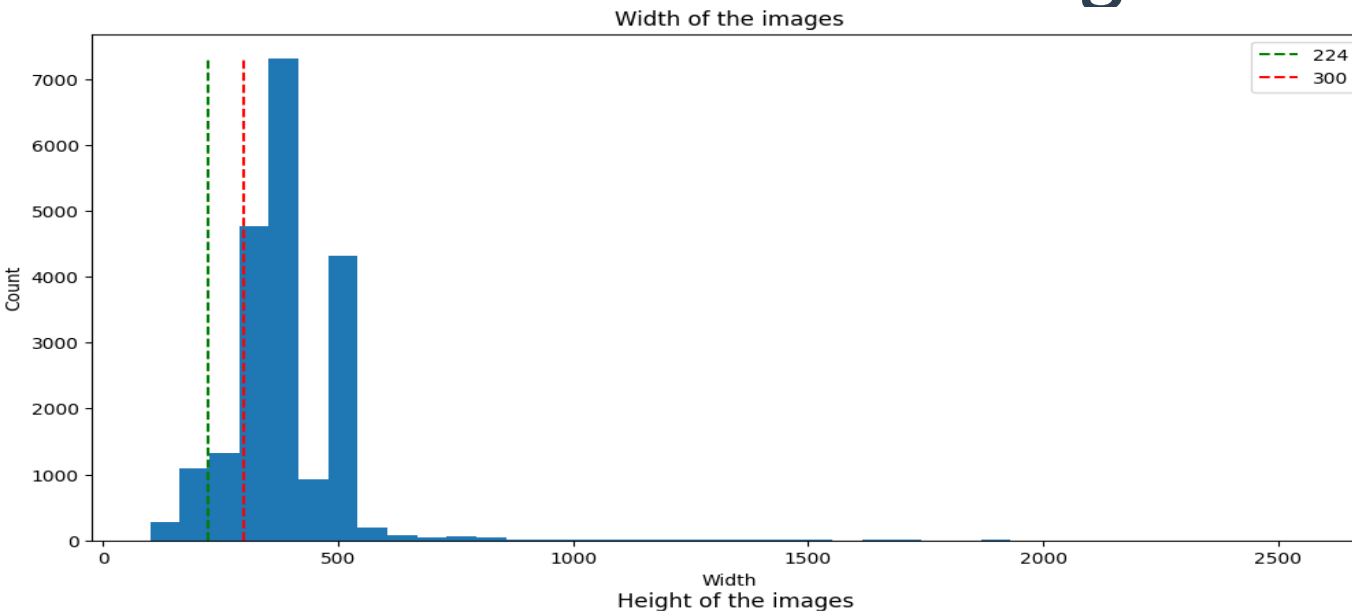
## Ratio hauteur / largeur du jeu d'images

Ratio of the dog images

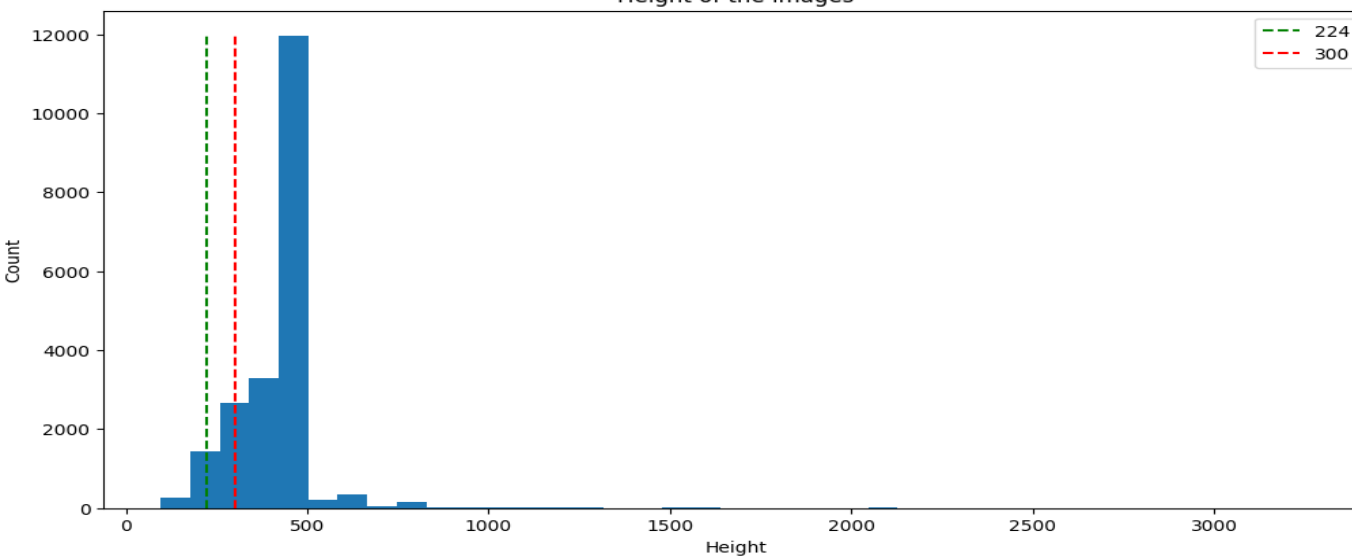


# Exploration des données

## Étude de la taille des images



Number of images bigger than 224x224:  
18987  
92.3 % of the whole dataset



Number of images bigger than 300x300:  
16857  
81.9 % of the whole dataset

# Exploration des données

## Choix du redimensionnement

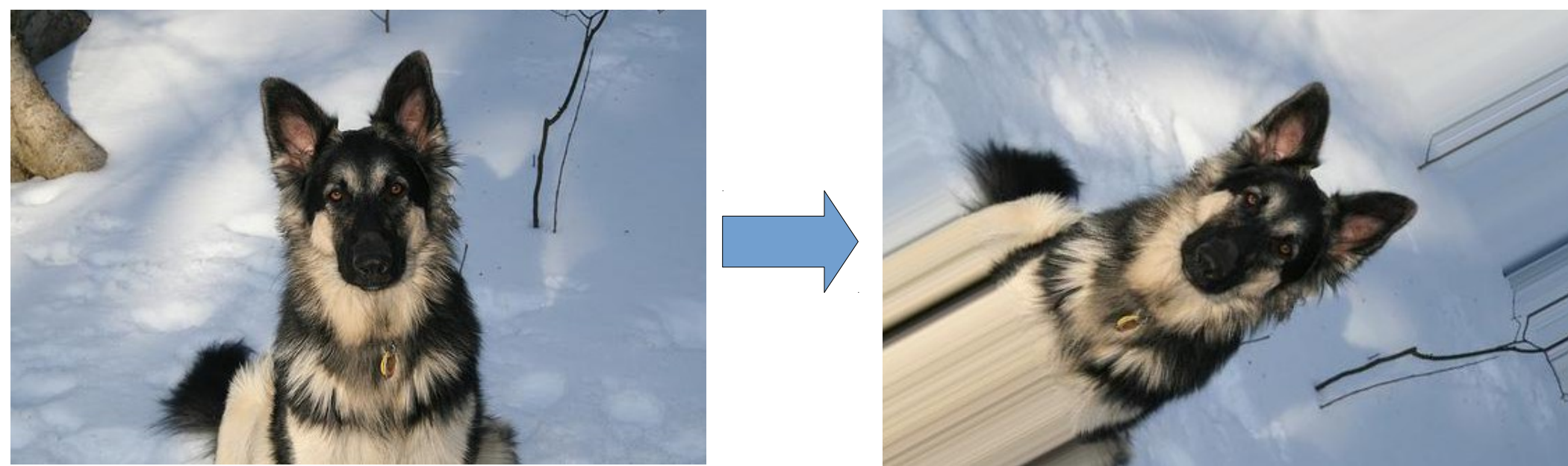
Redimensionnement	Utilisation	Jusitification
<i>Padding</i>	Non	Nombre de pixels suffisant
<i>Resize</i>	Oui	Réduction de taille Ratios équilibrés
<i>Cropping</i>	Non	Ratios non extrêmes

# Feature engineering

- **Redimensionnement** des images
- **Normalization** des images
- **Data augmentation** :
  - Rotation
  - Translation verticale
  - Translation horizontale
  - Inversion horizontale (*horizontal flip*)

# Feature engineering

## Data augmentation : Rotation



# Projet 6 : Soutenance

## Création d'un CNN

# Création d'un CNN

- Création des **jeux d'entraînement**, de **validation** et de **test** :
  - Répartition des photos grâce à *split-folders*
  - Création de jeux contenant seulement **10 classes**
  - **Redimensionnement** (150x150) et **normalization**
  - **Data augmentation** (*ImageDataGenerator*)
- Comparaison d'architectures :
  - AlexNet      `loss: 2.1334 - accuracy: 0.2086`
  - VGG 16      `loss: 2.2906 - accuracy: 0.1400`
- Callback utilisé : **early stopping**
- Choix de l'architecture de base : **AlexNet**



# Création d'un CNN

- **Modification de l'architecture AlexNet :**
  - Ajout de couches **Batch Normalization** après chaque couche cachée
  - Régularisation **l2** pour chaque couche FC
  - Ajout de couches **Dropout** après chaque couche FC
- **Compilation du modèle :**
  - Optimiseur : **Adam**
  - Perte : **categorical\_crossentropy**
  - Metric : **accuracy**
- **Callbacks :**
  - **Early stopping**
  - TensorBoard

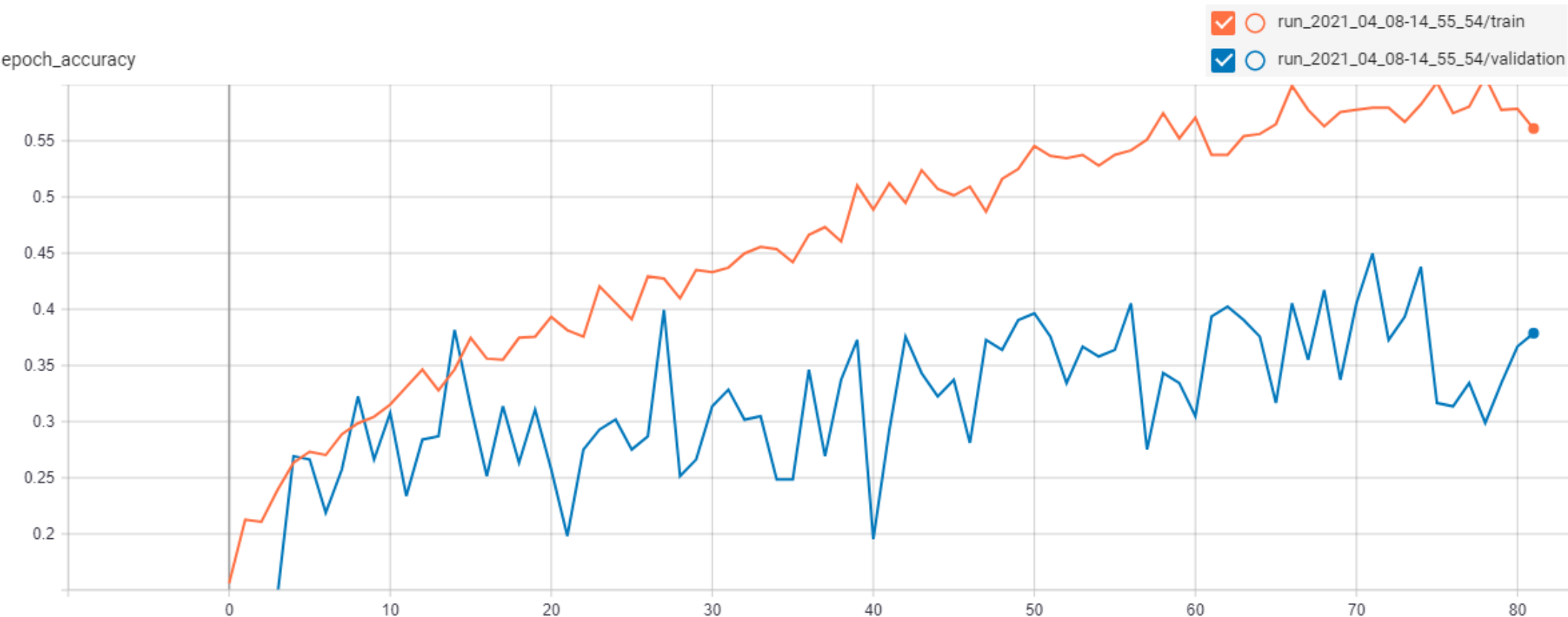
# Création d'un CNN

- Stratégie d'optimisation des hyperparamètres :
  - Optimisation **bayésienne** (*keras-tuner*)
- Hyperparamètres optimisés :
  - Taille du **noyau** de la **1ère couche** de convolution
  - Nombre de **filtres** pour chaque couche de convolution
  - Nombre de **neurones** pour chaque couche FC
  - **Taux d'extinction** pour chaque couche Dropout
  - **Taux d'apprentissage** de l'optimiseur Adam

# Création d'un CNN

## Performances du CNN

	Train set	Valid. set	Test set
Loss	2,68	3,31	3,36
Acc.	63,70 %	44,97 %	43,71 %



# Projet 6 : Soutenance

## Utilisation du *Transfer Learning*

# Utilisation du transfer learning

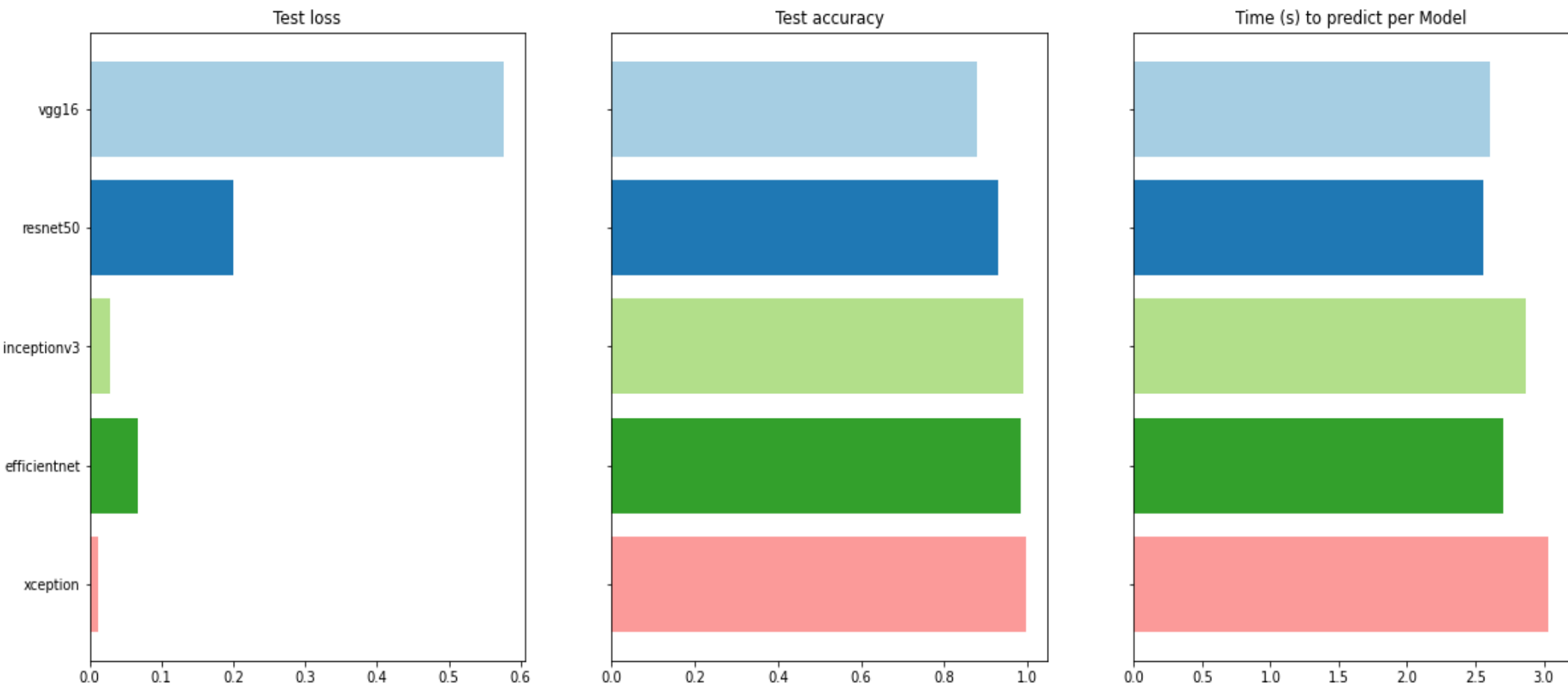
- Création des **jeux d'entraînement**, de **validation** et de **test** :
  - Création de jeux contenant seulement **10 classes**
  - **Redimensionnement** selon le modèle utilisé (224x224 ou 299x299)
  - Application **preprocess\_input** de chaque modèle (*keras.applications*)
  - **Data augmentation** (*ImageDataGenerator*)
- **Compilation** :
  - Optimiseur : **Adam**
  - Loss : **categorical\_crossentropy**
  - Metrics : **accuracy**
- **Callback utilisé** : **early stopping**

# Utilisation du transfer learning

- Modèles utilisés :
  - *VGG16*
  - *ResNet50*
  - *InceptionV3*
  - *EfficientNetB0*
  - *Xception*
- **Gèle** des **couches basses**
- Ajout couche **Global average pooling 2D**
- Ajout couche **FC** (10 unités, activation **softmax**)
- Fine tuning
- **Dégèle** des couches basses pour *VGG16* et *ResNet50* + fit (*lr* plus petit)

# Utilisation du transfer learning

## Choix du meilleur modèle



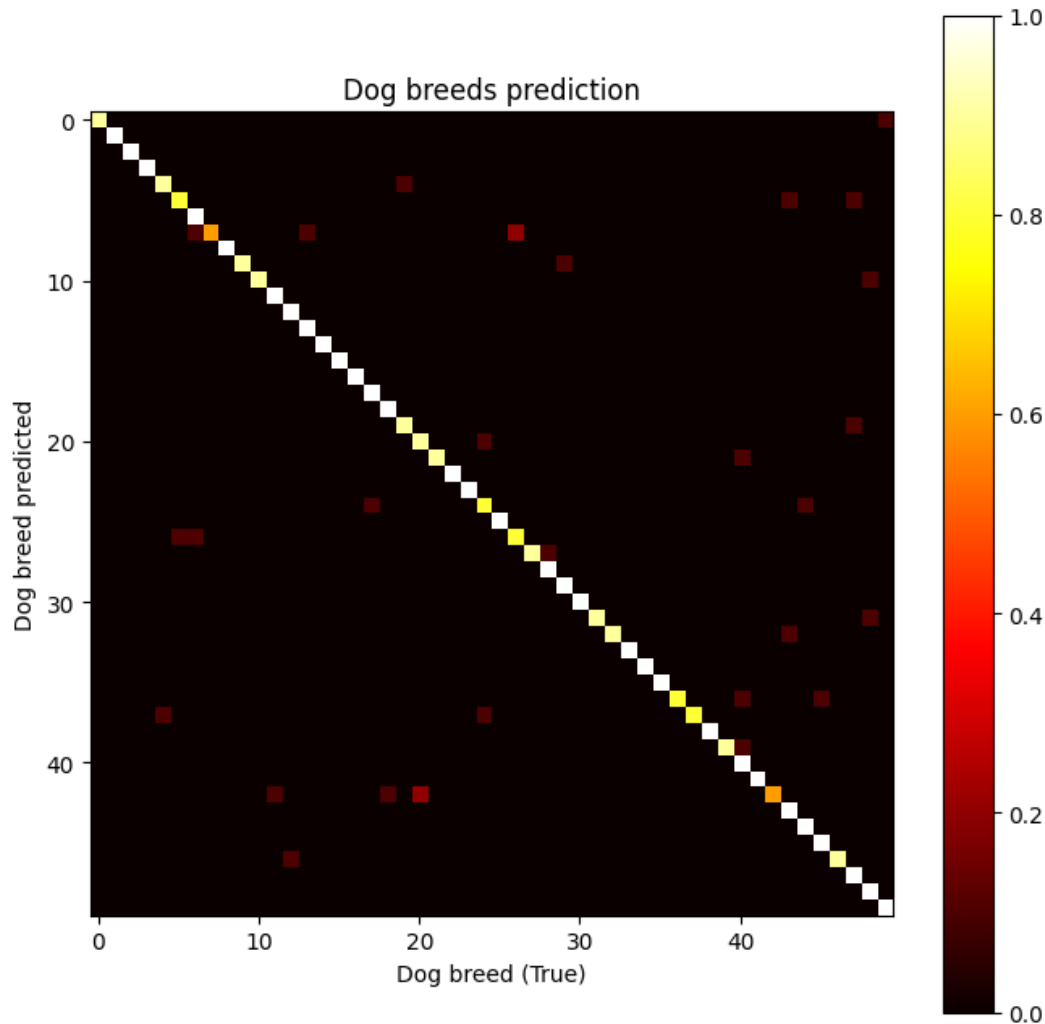
# Utilisation du transfer learning

- Modèle choisi → **Xception**
- Amélioration de l'algorithme avec plus de races :
  - Création de jeux contenant **50 classes**
  - **Redimensionnement** 299x299 et **preprocessing** pour le modèle Xception
  - **Data augmentation** (*ImageDataGenerator*)
- Ajout d'une couche de **Dropout**  
(coefficient d'extinction optimisé grâce à une recherche bayésienne)
- Couche FC (**50** unités, activation **softmax**)
- Compilation :
  - Optimiseur : **Adam**
  - Loss : **categorical\_crossentropy**
  - Metrics : **accuracy**
- Callback utilisé : **early stopping**



# Utilisation du transfer learning

## Xception adapté à 50 classes



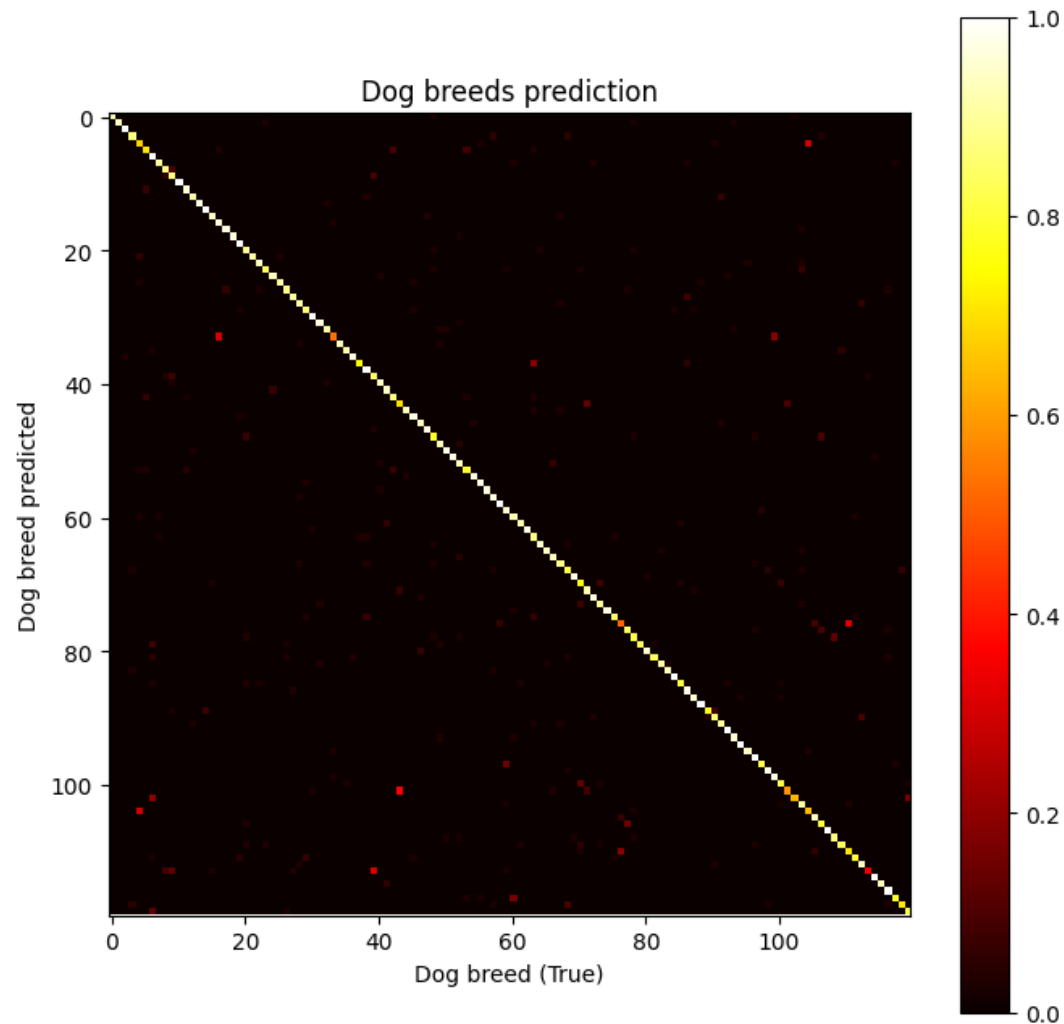
Modèle	Loss	Accuracy
Xception	0,1480	94 %

# Utilisation du transfer learning

- Modèle choisi → **Xception**
- Amélioration de l'algorithme avec la totalité des races :
  - Création de jeux contenant **120 classes**
  - **Redimensionnement** 299x299 et **preprocessing** pour le modèle Xception
  - **Data augmentation** (*ImageDataGenerator*)
- Ajout d'une couche de **Dropout**
- Couche FC (**120** unités, activation **softmax**)
- Compilation :
  - Optimiseur : **Adam**
  - Loss : **categorical\_crossentropy**
  - Metrics : **accuracy**
- Callback utilisé : **early stopping**

# Utilisation du transfer learning

## Xception adapté à 120 classes



Modèle	Loss	Accuracy
Xception	0,3385	89,31 %

# Projet 6 : Soutenance

## Déploiement d'une application

# Déploiement d'une application

- Utilisation de la librairie **Gradio**
- Dépôt des fichiers de l'application sur **Github**
- Déploiement de l'application avec Gradio Hosted :  
[https://gradio.app/g/GaetanPelletier/Dog\\_breeds\\_DL](https://gradio.app/g/GaetanPelletier/Dog_breeds_DL)

# Projet 6 : Soutenance

Synthèse

# Synthèse

- Analyse des données et feature engineering
  - Redimensionnement et normalisation
  - Data augmentation
- Modèle CNN from scratch
  - Utilisation d'un jeu de donnée réduit (10 classes)
  - Architecture AlexNet enrichie (Dropout, pénalité l2)
  - Optimisation d'hyperparamètres (recherche bayésienne)
  - Précision : 43,71 %
- Modèles CNN *transfer learning* :
  - Comparaison de cinq modèles
  - Entraînement du modèle final : Xception (enrichie d'une couche Dropout)
  - Précision (120 classes) : 89,31 %
- Déploiement du modèle dans une application web

## Projet 6 : Soutenance

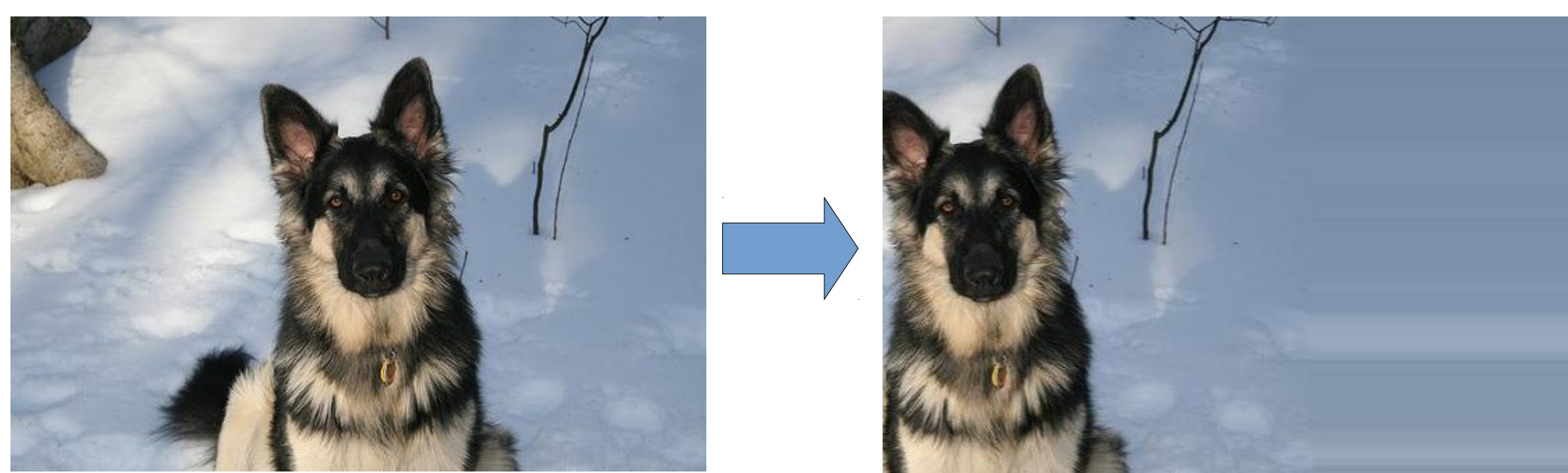
Merci de votre attention



## Annexes

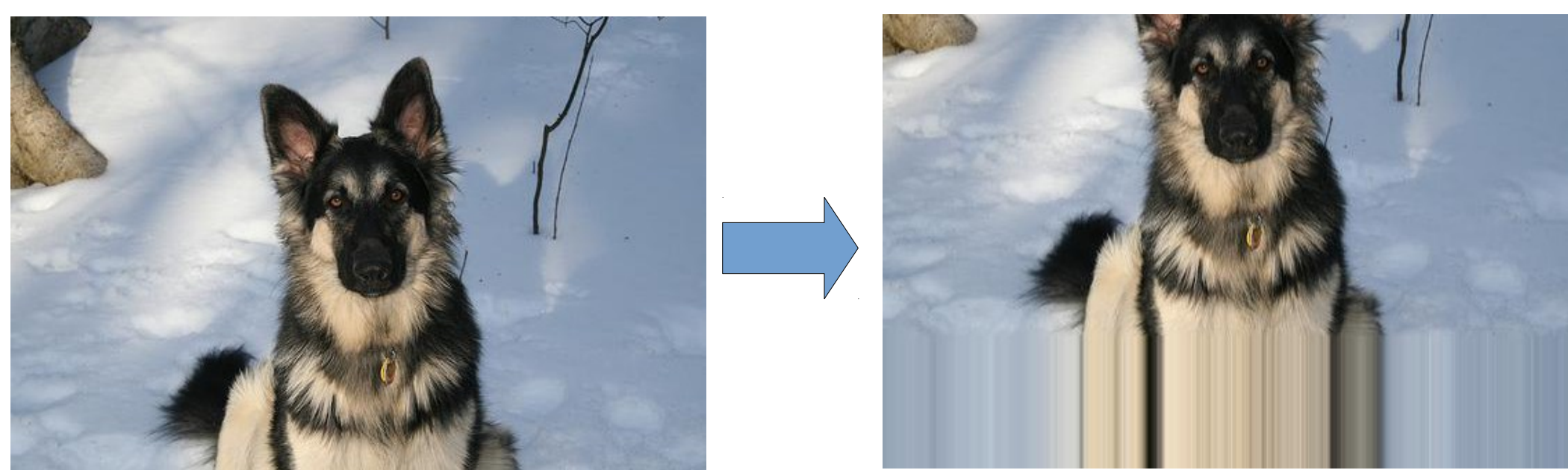
# Annexe 1

## Data augmentation : Translation horizontale



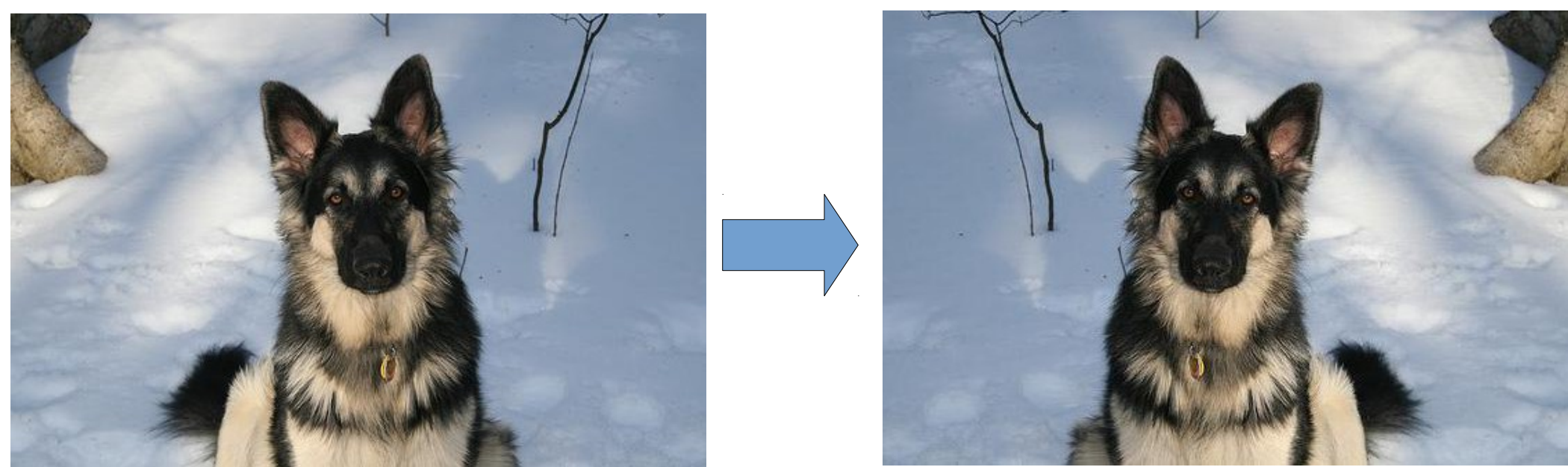
## Annexe 2

### Data augmentation : Translation verticale



## Annexe 3

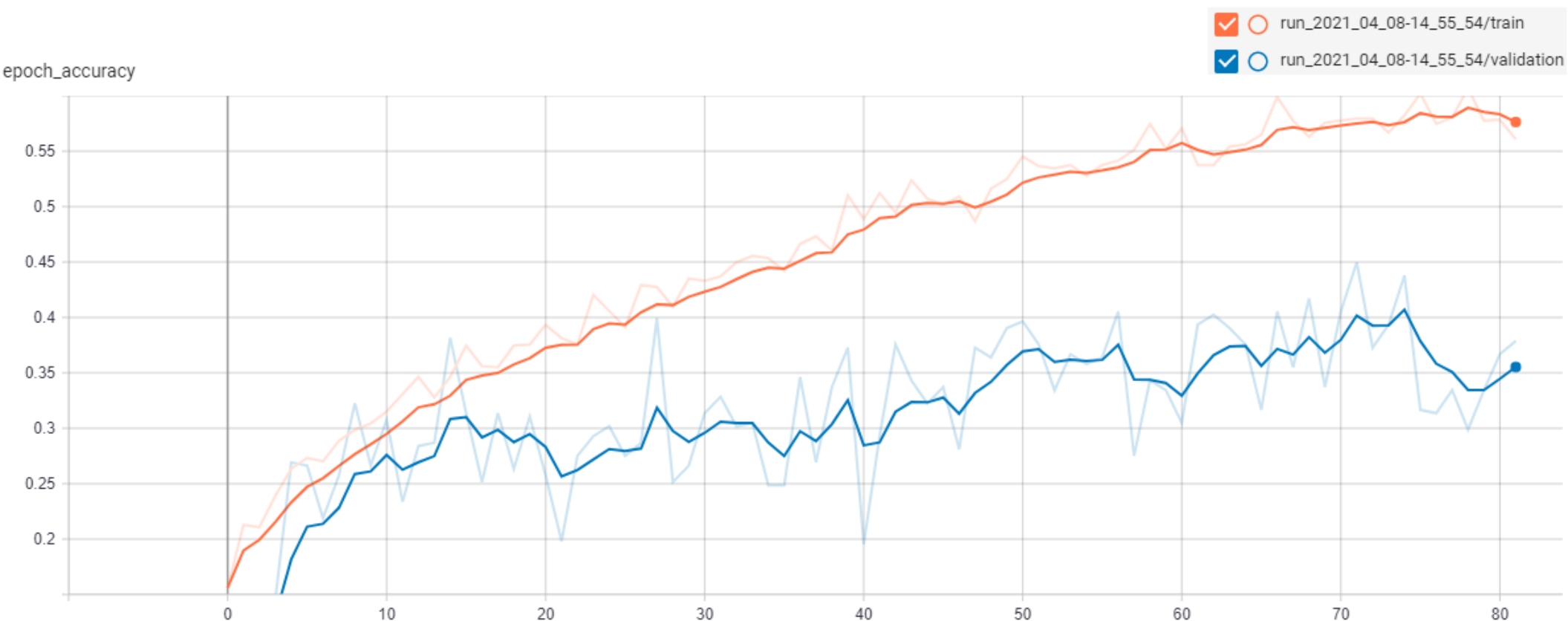
### Data augmentation : Inversion horizontale



# Annexe 4

## Performances du CNN (*smoothing* = 0,6)

	Train set	Valid. set	Test set
Loss	2,68	3,31	3,36
Acc.	63,70 %	44,97 %	43,71 %



# Création d'un CNN

## Architecture du CNN (à mettre en annexe?)

Layer (type)	Output Shape	Param #						
=====								
input_1 (InputLayer)	[(None, 150, 150, 3)]	0	conv2d_2 (Conv2D)	(None, 37, 37, 64)	147456	dense (Dense)	(None, 2048)	42467328
random_crop (RandomCrop)	(None, 150, 150, 3)	0	batch_normalization_2 (Batch Normalization)	(None, 37, 37, 64)	256	batch_normalization_5 (Batch Normalization)	(None, 2048)	8192
conv2d (Conv2D)	(None, 150, 150, 64)	9408	activation_2 (Activation)	(None, 37, 37, 64)	0	activation_5 (Activation)	(None, 2048)	0
batch_normalization (Batch Normalization)	(None, 150, 150, 64)	256	conv2d_3 (Conv2D)	(None, 37, 37, 256)	147456	dropout (Dropout)	(None, 2048)	0
activation (Activation)	(None, 150, 150, 64)	0	batch_normalization_3 (Batch Normalization)	(None, 37, 37, 256)	1024	dense_1 (Dense)	(None, 2048)	4194304
max_pooling2d (MaxPooling2D)	(None, 75, 75, 64)	0	activation_3 (Activation)	(None, 37, 37, 256)	0	batch_normalization_6 (Batch Normalization)	(None, 2048)	8192
conv2d_1 (Conv2D)	(None, 75, 75, 256)	147456	conv2d_4 (Conv2D)	(None, 37, 37, 64)	147456	activation_6 (Activation)	(None, 2048)	0
batch_normalization_1 (Batch Normalization)	(None, 75, 75, 256)	1024	batch_normalization_4 (Batch Normalization)	(None, 37, 37, 64)	256	dropout_1 (Dropout)	(None, 2048)	0
activation_1 (Activation)	(None, 75, 75, 256)	0	activation_4 (Activation)	(None, 37, 37, 64)	0	dense_2 (Dense)	(None, 10)	20490
max_pooling2d_1 (MaxPooling2D)	(None, 37, 37, 256)	0	max_pooling2d_2 (MaxPooling2D)	(None, 18, 18, 64)	0	=====		
			flatten (Flatten)	(None, 20736)	0	Total params: 47,300,554		
						Trainable params: 47,290,954		
						Non-trainable params: 9,600		



# Annexe 5

## Exemple de rapport de classification (120 classes)

	precision	recall	f1-score
Dingo	0.86	1.00	0.92
English_springer	0.66	0.84	0.74
Dhole	0.91	0.94	0.93
Doberman	0.89	0.94	0.91
EntleBucher	0.84	0.88	0.86
Eskimo dog	0.62	0.70	0.66
Flat-coated_retriever	0.97	0.94	0.95
French_bulldog	0.91	0.97	0.94
German_short-haired_pointer	0.91	0.94	0.92
German_shepherd	0.97	0.97	0.97
Gordon_setter	0.81	0.78	0.79
Great_Pyrenees	0.88	0.93	0.90
Greater_Swiss_Mountain_dog	0.94	0.97	0.95
Ibizan_hound	1.00	0.97	0.98
Great_Dane	0.84	0.93	0.88
Groenendael	0.85	0.80	0.82
Golden retriever	0.97	0.97	0.97




# Annexe 6

## Application :

### Image classification - 120 dog breeds

IMAGE



EDIT

OUTPUT

## Siberian\_husky

Siberian_h...	65%
Eskimo_dog	34%
Malamute	0%

Latency: 0.15s

CLEAR

SUBMIT

SCREENSHOT

GIF

FLAG

#### Examples

Run All

Load Previous  
CTRL + ←

Load Next  
CTRL + →

≡

■ ■

