Développement d'une preuve de concept

Présenté par : SEKPONA Kokou Sitsopé,
Etudiant en ingénierie Machine Learning chez Openclassrooms/ Central Supélec
Date : 12 Décembre 2022

Introduction

La classification d’images est un sujet très ancien dans l’histoire du Deep Learning.

Mais le problème de classification est complexe en soi, où plusieurs paramètres sont à prendre en compte, donc on ne peut qu’essayer d’améliorer les performances pour être plus proche de la réalité au lieu de chercher à avoir un modèle parfait. Face à cette inquiétude, plusieurs hypothèses et algorithmes ont été énoncés.

Comment donc améliorer les performances d’un modèle de classification d’images

Aperçu du Projet

L'association de notre quartier pour lequel nous avons travaillé dans le projet 6 a remarqué que le model fait parfois de mauvaises prédictions à cause du faible accuracy du modèle utilisé. Nous allons donc trouver des techniques permettant d'améliorer les performances de cette classification en nous basant sur de nouvelles méthodes.

Recherche d'algorithmes : Sources consultées

Vision Transformers provably learn spatial structure

Renseignement sur la performance des ViT.
https://arxiv.org/abs/2210.09221

Understanding the Vision Transformer and Counting Its Parameters

Explication mathématique des ViT
https://medium.com/analytics-vidhya/understanding-the-vision-transformer-and-counting-its-parameters-988a4ea2b8f3

Image classification with Vision Transformer

Implémentation des ViT en python(site de karas)

https://keras.io/examples/vision/image_classification_with_vision_transformer/

Fine-Tune Visual Transformers for Image Classification with Transformers/ Kaggle Notebook

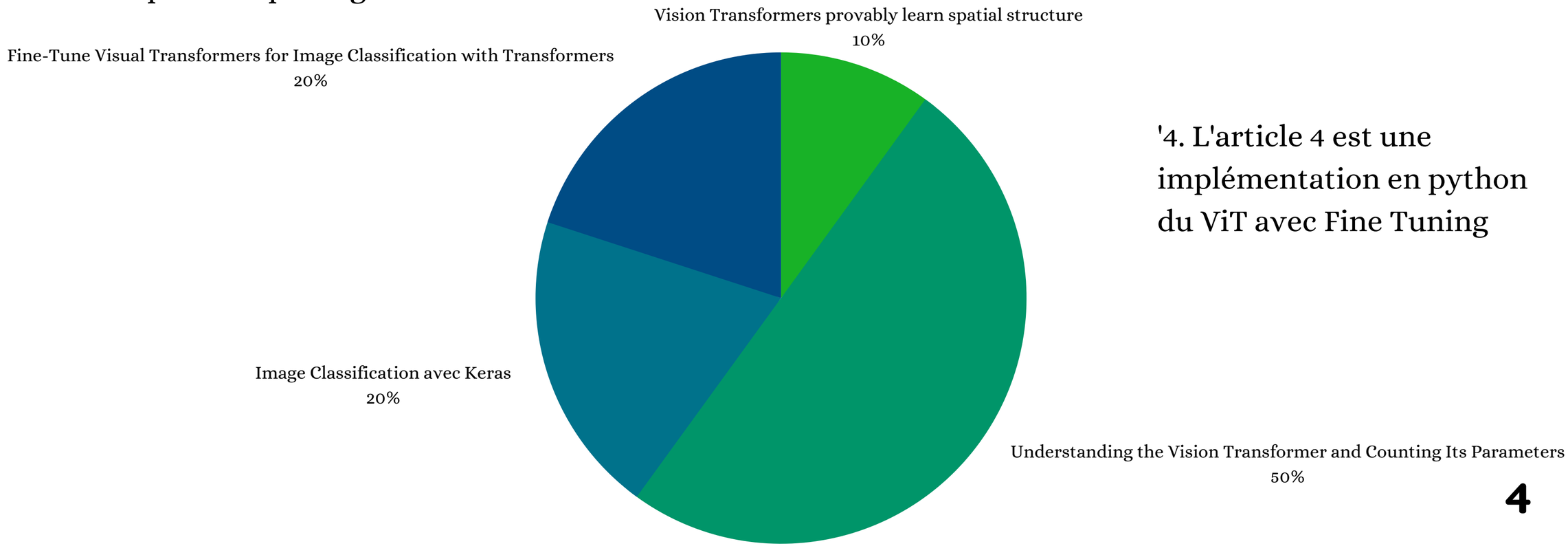
Implémentation en python du ViT B32 avec fine tuning
https://huggingface.co/blog/fine-tune-vit

Detail sur les articles

1. L'article affirme clairement qu'en général, les ViT ont une bonne performance par rapport aux CNN et aux réseaux existants. Mais comme cet article ne contient pas de code, il n'a servi qu'a nous spécifier quel algorithme utiliser.

2.Le Deuxième article explique l'algorithme et les formules mathématiques sur lesquels il est basé

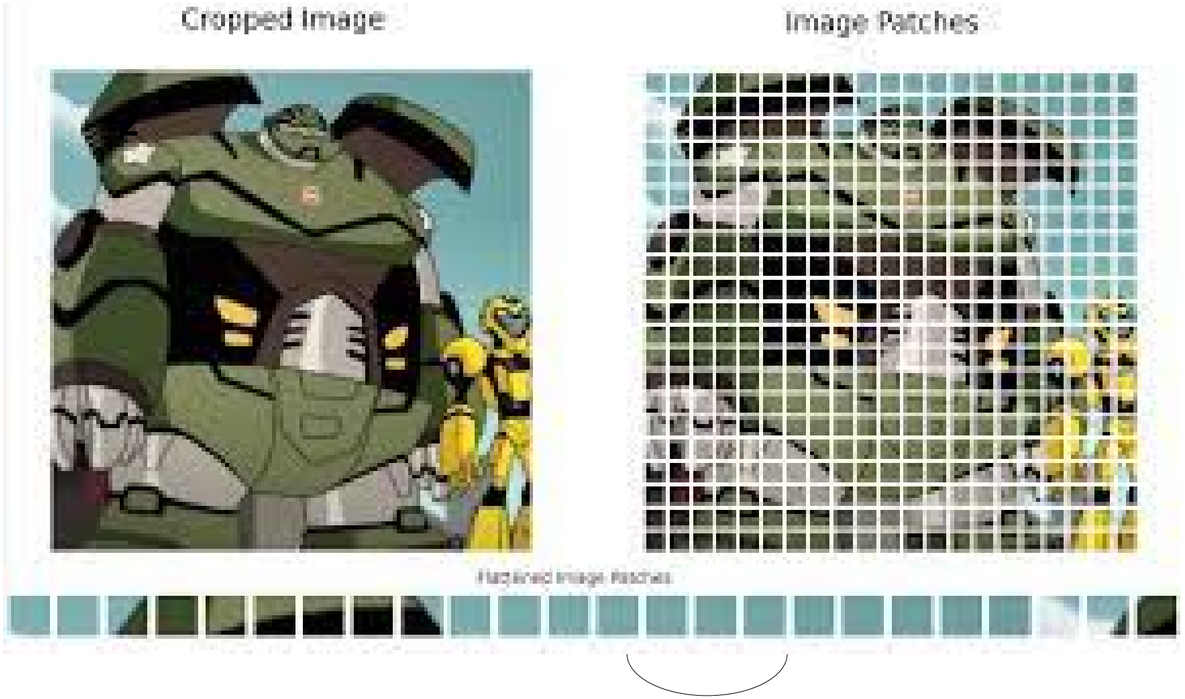
3. L'article 3 est une implémentation du ViT en python sur le dataset de cifar 100.



'4. L'article 4 est une implémentation en python du ViT avec Fine Tuning

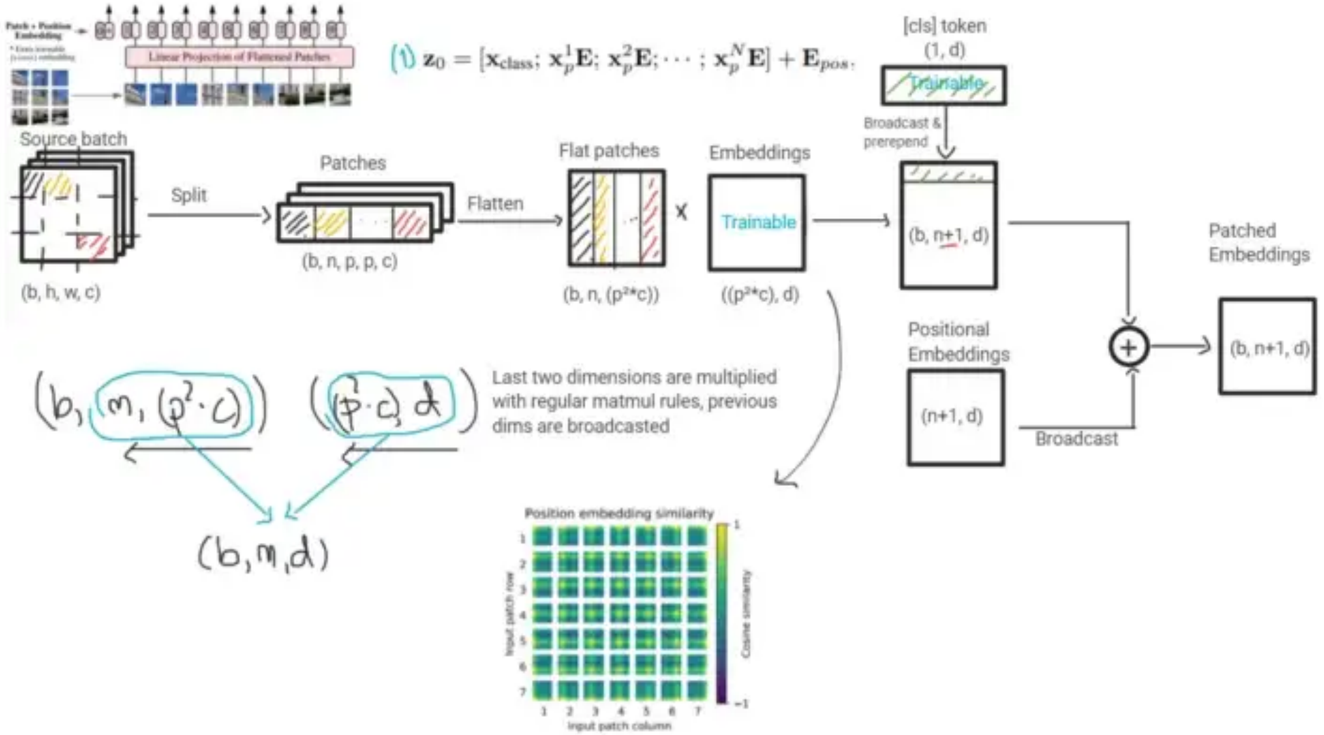
Le Visual Transformers

Basé sur le fameux article "Attention is All We need" , le ViT est un dérivé des Transformers qui à la base, sont plus développés pour le traitement de langage naturel.
Le traitement d'image utilisait du CNN
Mais le calcul des relations pour chaque paire de pixels dans une image typique est prohibitif en termes de mémoire et de calcul. Au lieu de cela, ViT calcule les relations entre les pixels dans diverses petites sections de l'image (par exemple, 16x16 pixels), à un coût considérablement réduit.



Explication de l'Alorithme des ViT

1. Patch Embedding

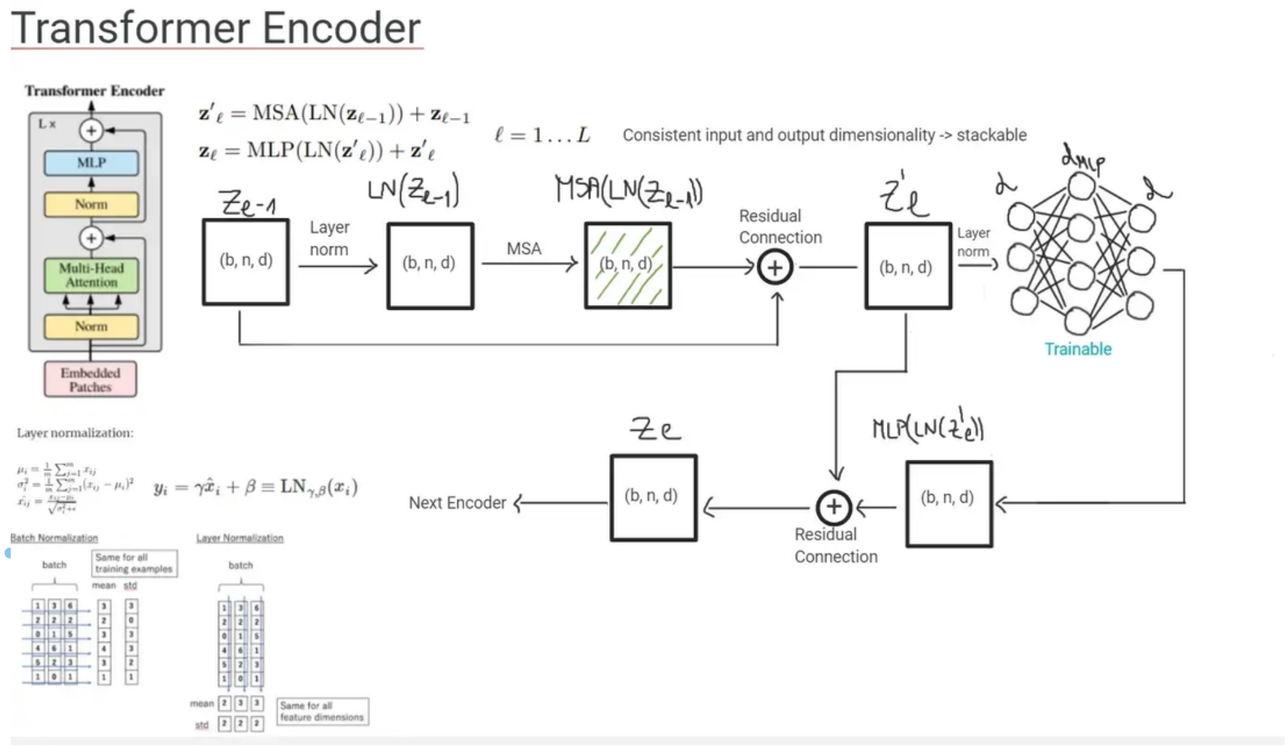


une image d'entrée de forme *(hauteur, largeur, channel)* est intégrée dans un vecteur caractéristique de forme *(n+1, d)* , suivant une séquence de transformations. Ce qui est décrit par la formule suivante:

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\ \mathbf{x}_p^1\mathbf{E};\ \mathbf{x}_p^2\mathbf{E};\ \cdots\ ;\ \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \qquad (1)$$

2. Transformer Encoder

Le réseau apprend plus de fonctionnalités abstraites à partir des patchs intégrés, en utilisant une pile d'encodeurs par transformateur , ce qui est décrit par les équations suivantes:



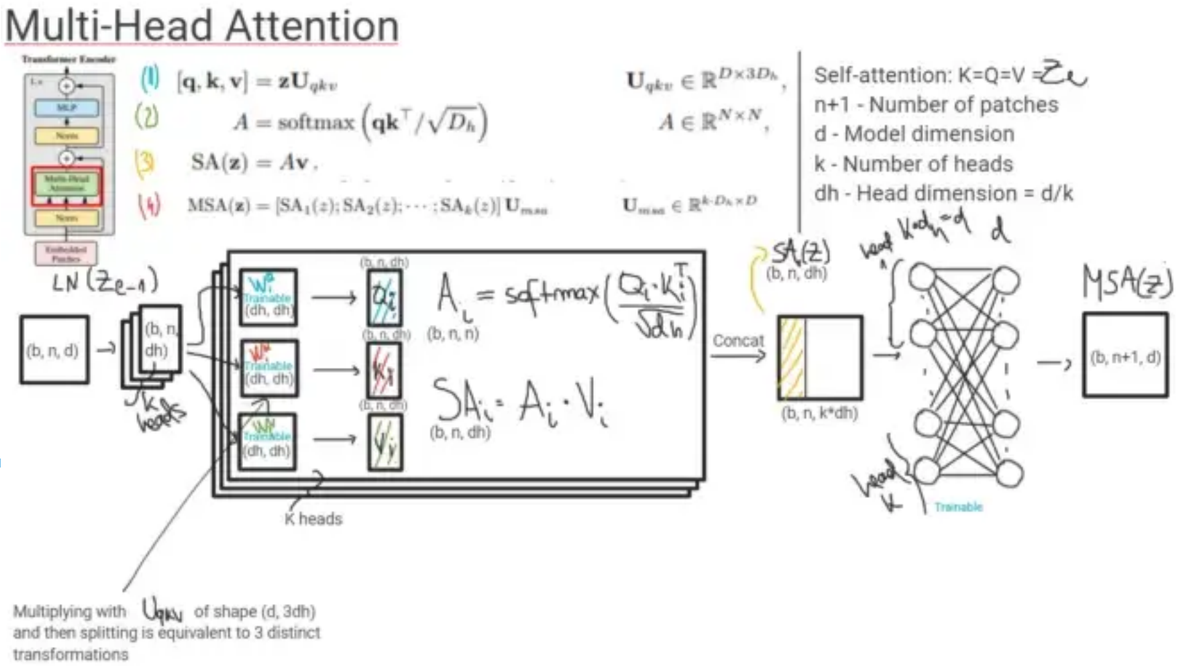
$$\mathbf{z'}_\ell = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1 \ldots L \qquad (2)$$

$$\mathbf{z}_\ell = \mathrm{MLP}(\mathrm{LN}(\mathbf{z'}_\ell)) + \mathbf{z'}_\ell, \qquad \ell = 1 \ldots L \qquad (3)$$

Detail des MultiHead Attention

Le composant codeur contient un mécanisme d'attention multi-têtes (MHA) et un MLP à 2 couches, avec une normalisation de couche

L'étape d'attention multi-tête (MHA), correspond aux équations suivantes



$$[\mathbf{q}, \mathbf{k}, \mathbf{v}] = \mathbf{z}\mathbf{U}_{qkv} \qquad \mathbf{U}_{qkv} \in \mathbb{R}^{D\times 3D_h}, \tag{5}$$

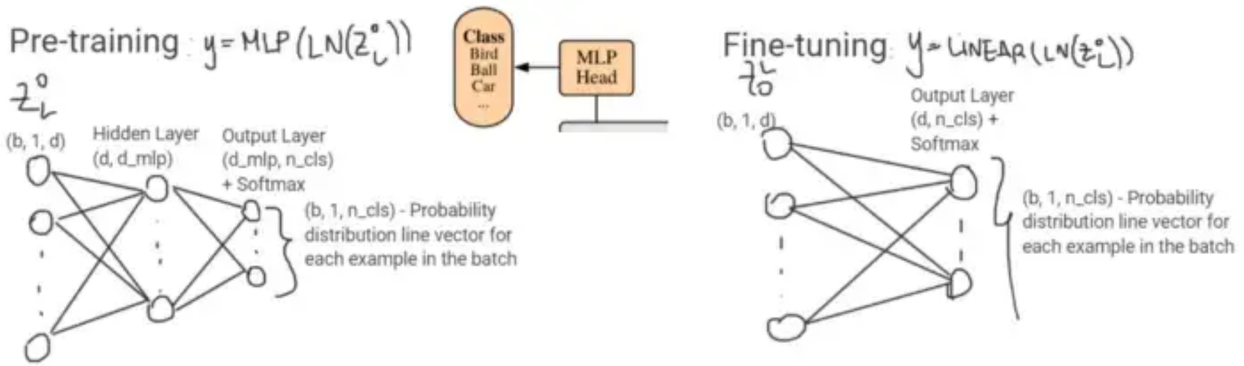
$$A = \text{softmax}\left(\mathbf{q}\mathbf{k}^\top/\sqrt{D_h}\right) \qquad A \in \mathbb{R}^{N\times N}, \tag{6}$$

$$\text{SA}(\mathbf{z}) = A\mathbf{v}\,. \tag{7}$$

$$\text{MSA}(\mathbf{z}) = [\text{SA}_1(z); \text{SA}_2(z); \cdots; \text{SA}_k(z)]\,\mathbf{U}_{msa} \qquad \mathbf{U}_{msa} \in \mathbb{R}^{k\cdot D_h \times D} \tag{8}$$

3. Etape de classification

Ici, le perceptron multicouches MLP est utilisé pour l'etape du pré entrainement et le Fine Tuning a une activation softmax lui permettant de produire une probabilité d'appartenance à chaque classe.

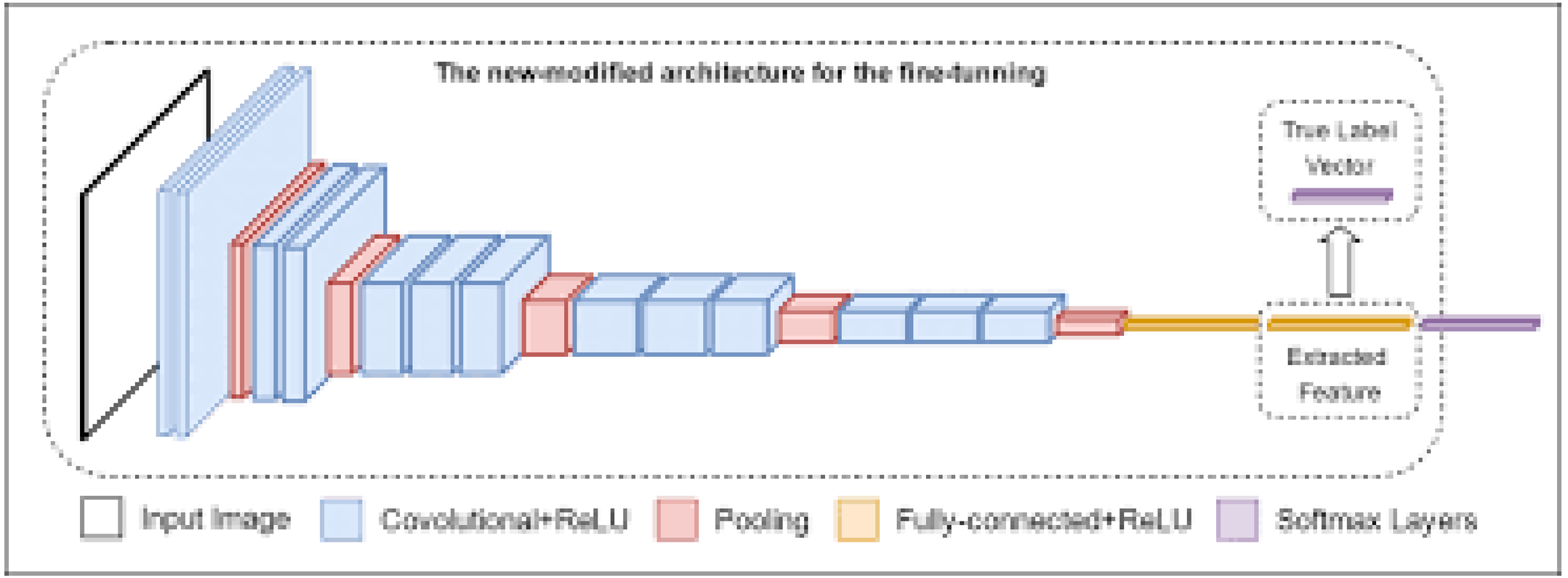


Dataset

Notre objectif étant d'améliorer les performances obtenus dans le projet précédent, nous allons donc travailler sur le meme dataset: le Standford Dogs DataSet, une base de données composé de 120 differentes races de chiens

Nous répartissions la dataset
en train et en test
et le train contient 100 images
par classes

Model Baseline: Le Vgg16 avec Fine Tuning



Puisque le model retenu pour l'association était le model vgg16 avec Fine-Tuning, Nous allons donc la considérer comme baseline et developper les models basés sur le Vit puis évaluer leur performance.

Data Augmentation

Images générés par zoom, flip, rotation ...

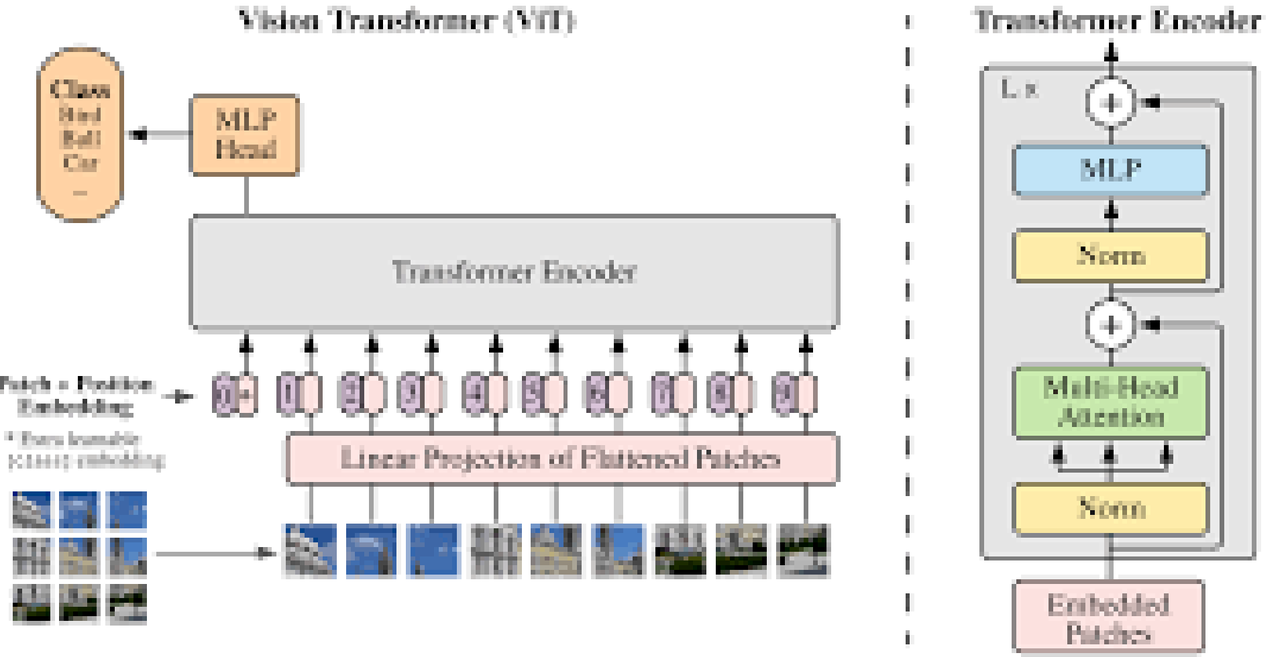
Prediction Transfert Learning(Vgg16 Fine Tuning)

input_3 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten_4 (Flatten)	(None, 25088)	0
dense_9 (Dense)	(None, 120)	3010680

Notre model a :

- **17.725.368 paramètres au total,**
- **3.010.680 paramètres pouvant être entraînés**
- **14.714.688 paramètres non entraînable**

Test loss: 8.028903
Test accuracy: 0.2286

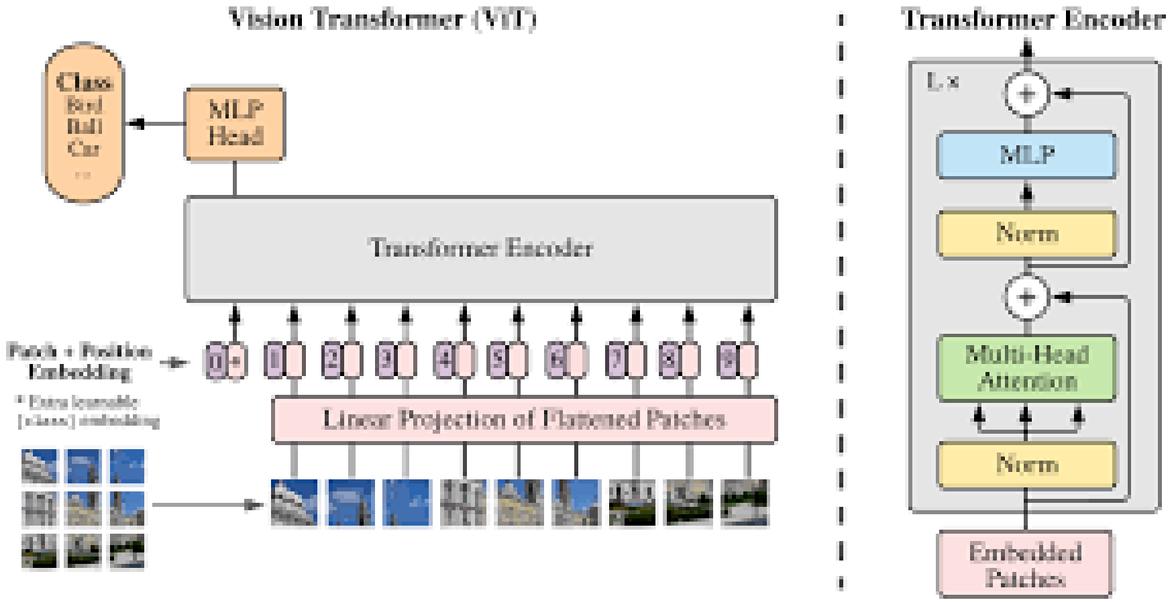


Implémentation du ViT

Visual Transformers

Notre ViT est faite de :

- **Une entrée (Input)**
- **De la Data augmentation**
- **Patches Encoding**
- **Du LayerNormalization**
- **Du MultiHeadAttention**
- **Du perceptron Multicouches**
- **D'une couche de Flatten**
- **D'une couche de Dropout**
- **Et d'une couche final de sortie**



Test de differents parametres pour comparer l'accuracy

D'apres la documentation de keras sur le Vit, on peux avoir de meilleur selon le type de dataset, mais aussi en agissant sur le nombre d'epochs ou l'input shape.
Nous allons agir sur ces parametres et noter la précision

- **Nombre d'epoch=100**
- **Input shape: 72, 72, 3**

Test accuracy: 0.89%

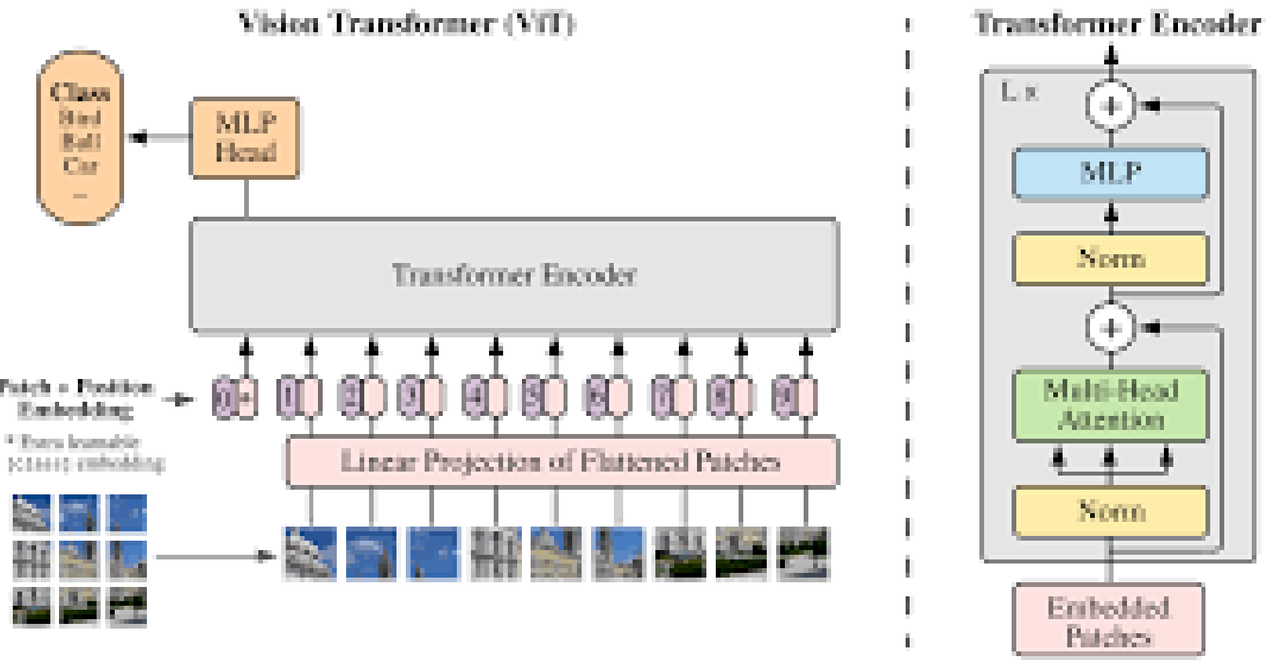
- **Nombre d'epoch=100**
- **Input shape: 32, 32, 3**

Test accuracy: 0.97%

- **Nombre d'epoch=200**
- **Input shape: 72, 72, 3**

Test accuracy: 0.86%

Nous avons obtenu une tres mauvaise précision,
la documentation du keras précise aussi que l'utilisation des models pré entrainés ViT peuvent améliorer la performance. Nous allons donc implémenter un: le ViT B32



Implémentation du Vision Transformer ViT B32 avec Fine-tuning

Quelques prétraitement et choix des parametres

Encodage des classes allant de 0 à 119

Définition des parametres:
IMAGE_SIZE = 224
BATCH_SIZE = 16
EPOCHS = 10

Data Augmentation:
Flip, Rotate, Rescaling

Images obtenus par data augmentation

Exemple of image patches

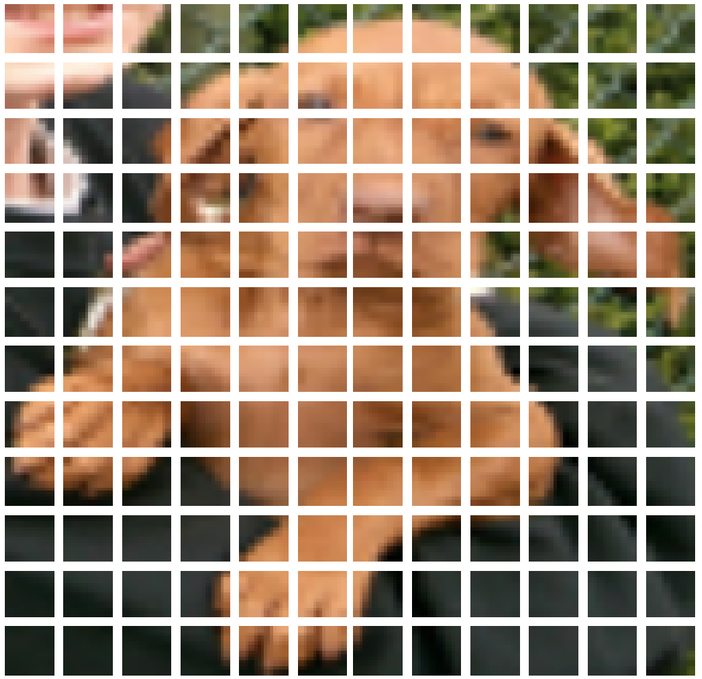


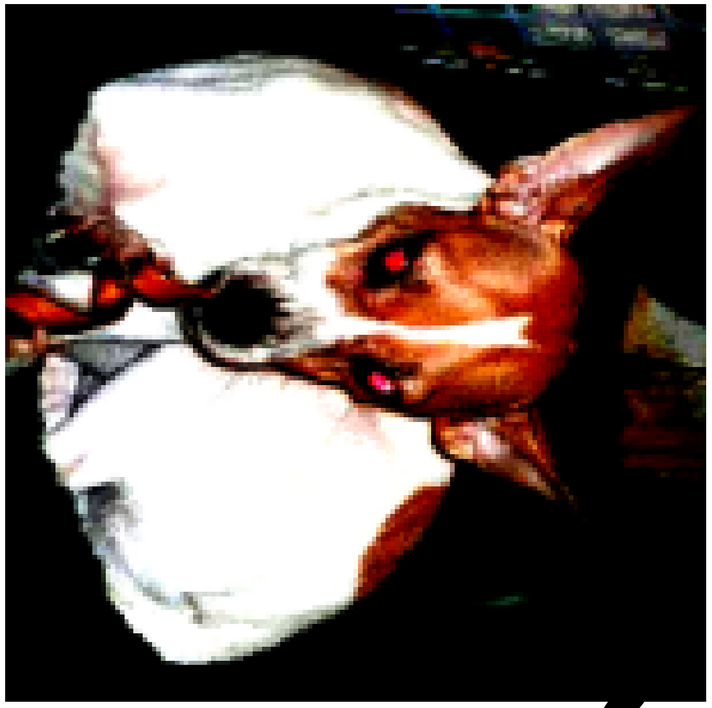
Image size: 72 X 72 Patch size:

6 X 6 Patches per image: 144

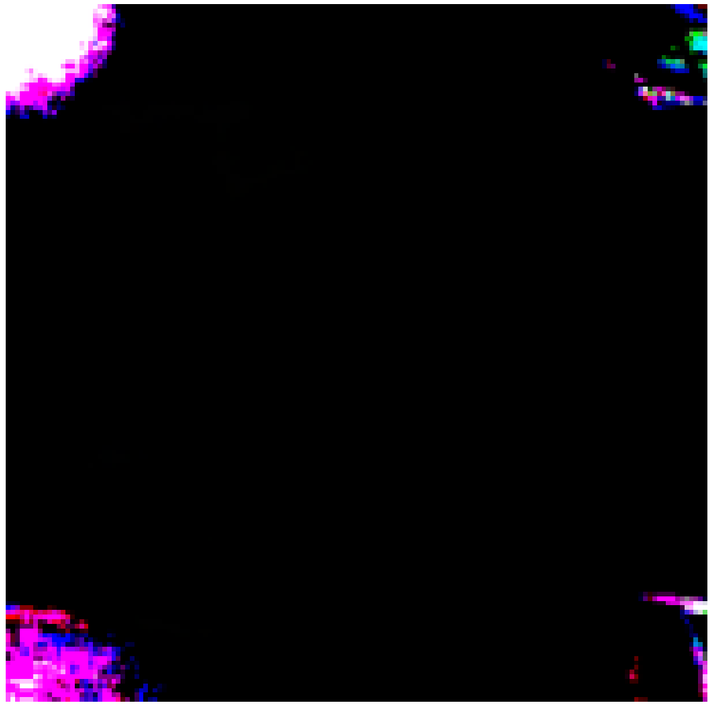
Elements per patch: 108

Visualisation d'Attentions Marks d'un exemple d'image

Original



Attention Map



Résumé du model

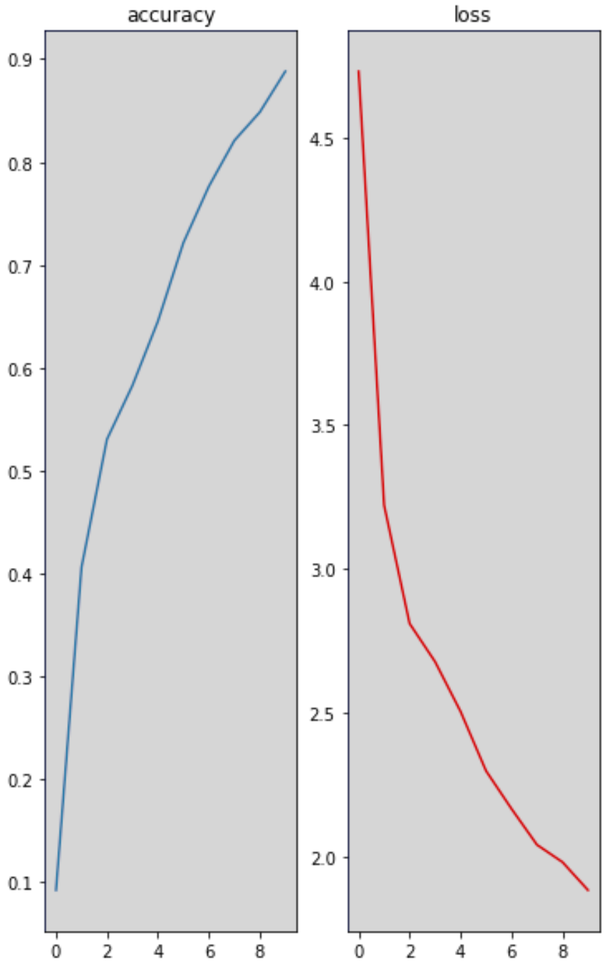
```
Model: "vision_transformer"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vit-b32 (Functional)         (None, 768)               87455232
_________________________________________________________________
flatten_4 (Flatten)          (None, 768)               0
_________________________________________________________________
batch_normalization_6 (Batch (None, 768)               3072
_________________________________________________________________
dense_26 (Dense)             (None, 512)               393728
_________________________________________________________________
batch_normalization_7 (Batch (None, 512)               2048
_________________________________________________________________
dense_27 (Dense)             (None, 120)               61560
=================================================================
```

Notre model a :

- 87.915.640 paramètres au total,
- 87.913.080 paramètres pouvant être entraînés
- 2.560 paramètres non entraînable

Nous entrainons le model sur 10 epochs

Evaluation et metrics

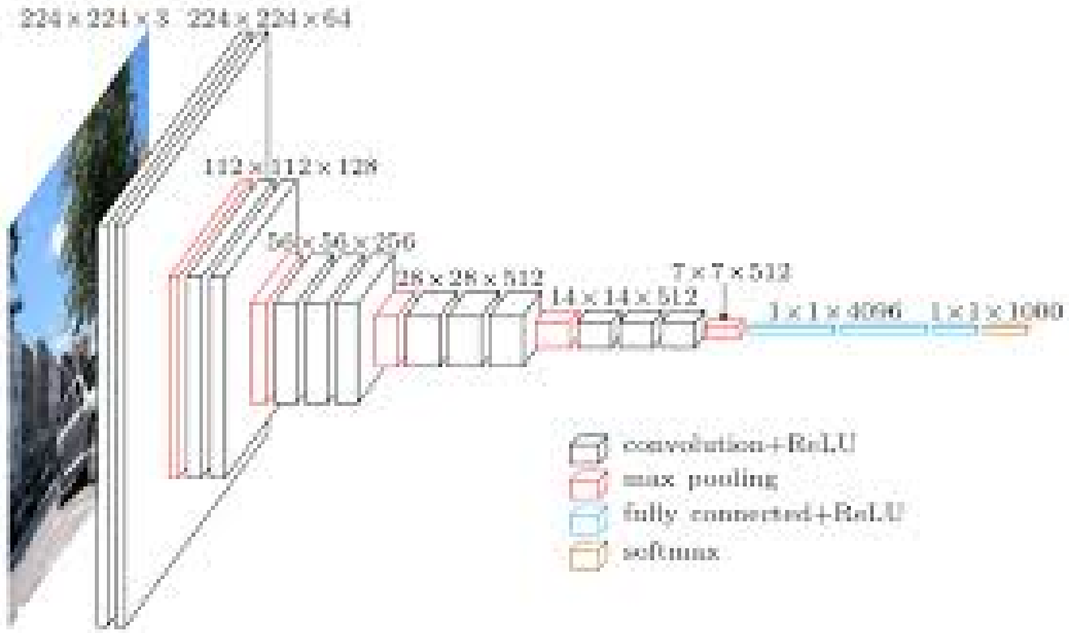


Test loss: 3.1284811462

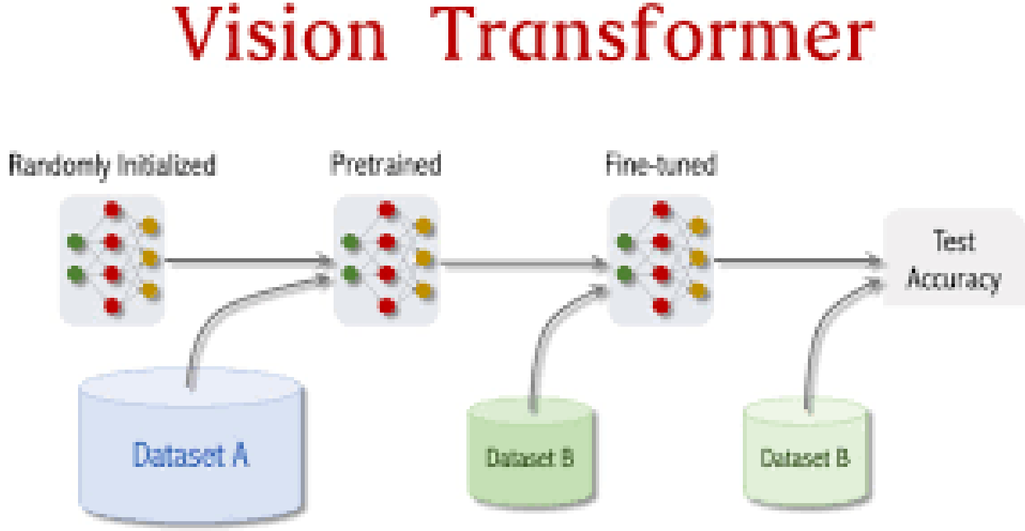
Test accuracy: 0.501165509

Notre model a un accuracy de 50% et un loss de 3.12

Comparaison des models et conclusion



? **VGG16 Fine Tuning**
Vs
ViT Fine-Tuning



Comparaison des 2 models

Le Vgg16 est plus rapide en entrainement que le ViT B32

Le Vit B32 a un test_accuracy 2 fois plus élevé et un test_loss deux fois moins réduit que le vgg16

Le vgg16 a beaucoup moins de parametres que le ViT B32

CONCLUSION

En conclusion, Nous retenons le ViT car plus précis, notre objectif final étant de pouvoir améliorer la classification.

Merci !