VISION TRANSFORMER

TMI

CONVOLUTION ET CHAMP FOCAL

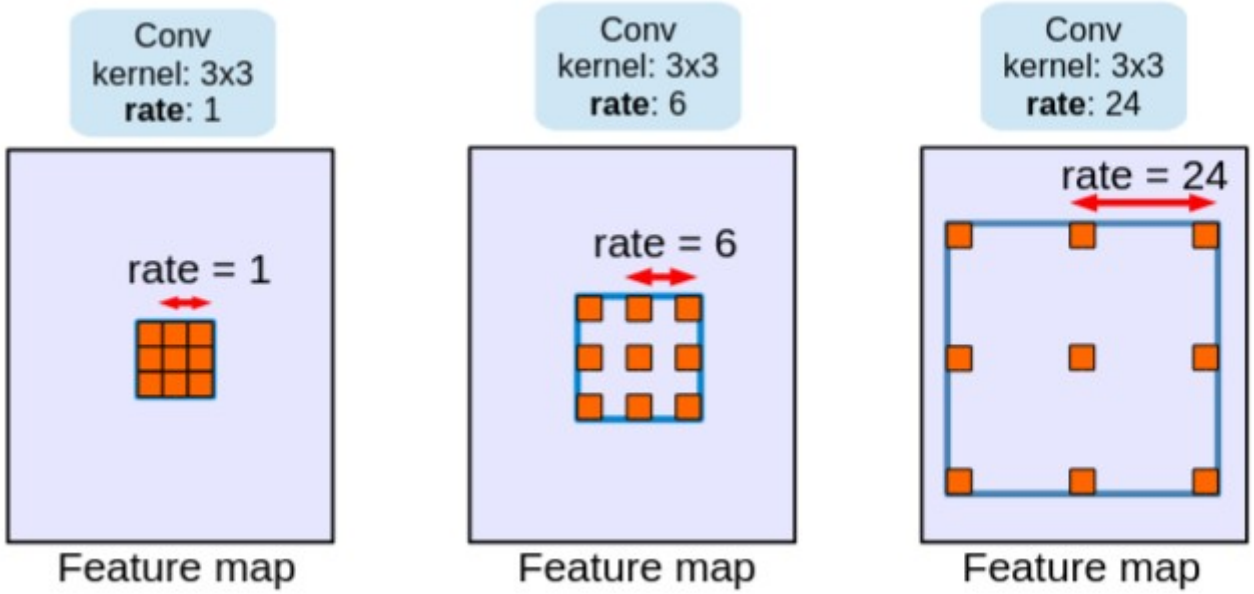
Convolution = analyse locale → champ focal réduit

Élargissement :

Gros noyau de convolution : lent, et limité (15x15 est déjà gros, 225 poids)

Atrous convolutions aussi appelées *dilated convolutions*

- 3x3=9 poids
- Perte de localité si trop gros
- Perte de détails



Extrait de « Rethinking atrous convolutions for semantic image segmentation », Chen et al. CoRR 2017.

CONVOLUTION ET CHAMP FOCAL

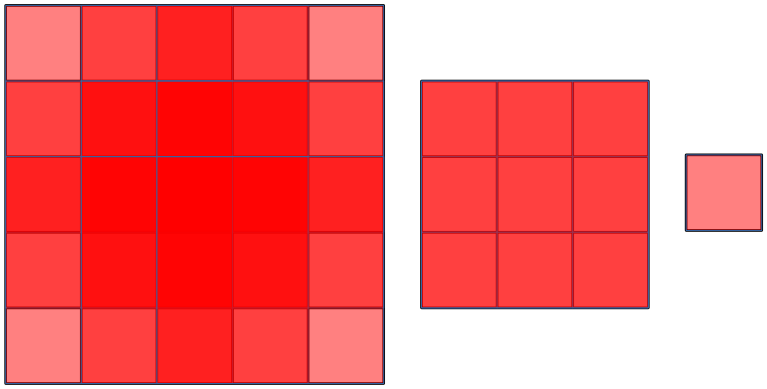
VGG

Empiler les petites convolutions

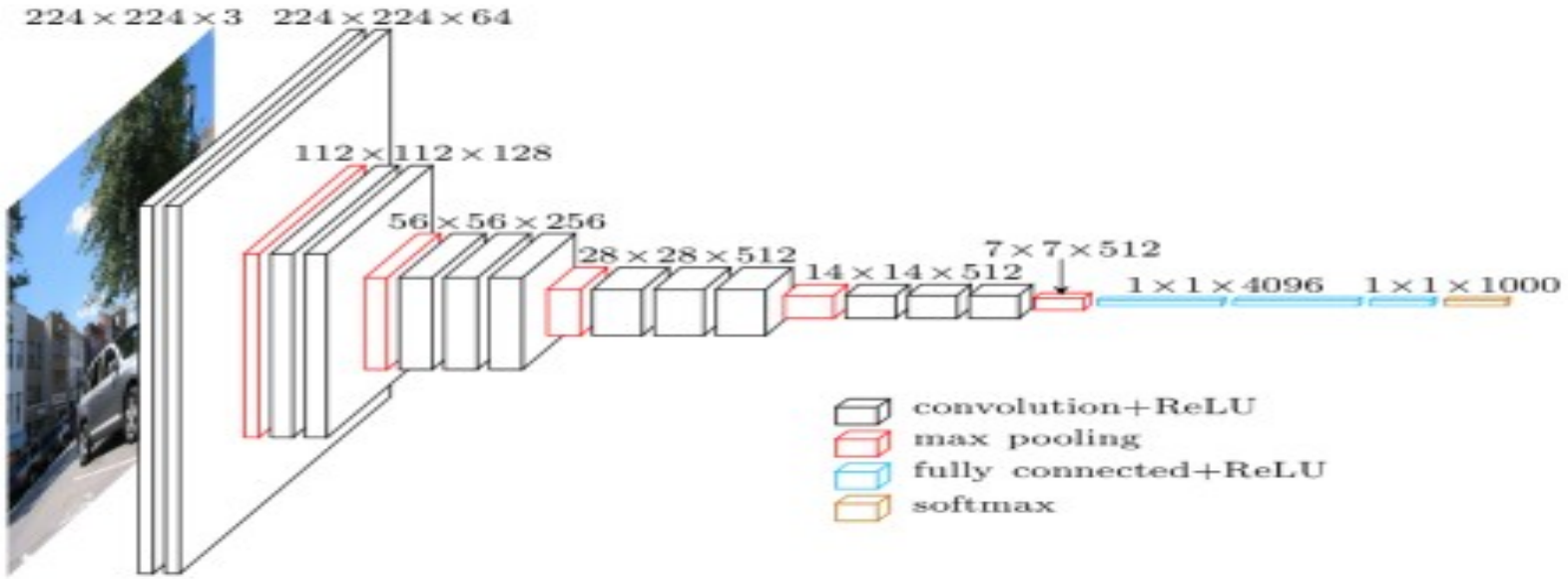
N convolutions 3x3 → champ focal 2N+1 x 2N+1

9xN paramètres

Ex : 63 pour un noyau 15x15 (N=7)



VGG-16



- Plus profond qu'AlexNet (plusieurs couches de convolution à suivre) : 138 millions de paramètres
- Parallélisme GPU (4) : batch splitting
- Convolutions minimales (3x3), maxpool sans overlap
- *Local response normalization* inutile et pèse sur mémoire
- Top-5 error : 7.32 % (ILSVRC 2014)
- Pas d'amélioration avec profondeur de 19 couches…

Very deep convolution networks for large-scale image recognition. Simonyan and Zisserman, *ICLR 2015*.

GoogLeNet (Inception modules)

Problème :

Objets peuvent être petits ou grands dans l'image
→ petits ou grand kernels, maxpool (stride 1)

Solution

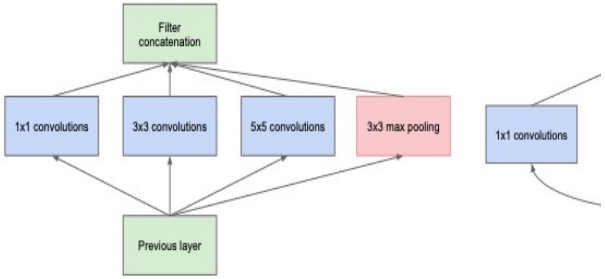
Plusieurs tailles de convolution + maxpool combinées

Ajoute réduction de dimension (convolution 1x1) pour réduire le coût de calcul

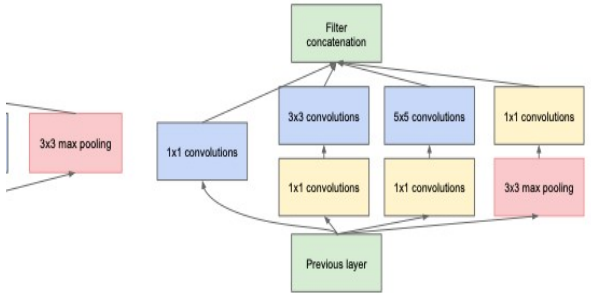
27 couches de réseau

Résultat

Top-5 error : 6.67 % (ILSVRC 2014)



(a) Inception module, naïve version (b) Inceptior



ersion (b) Inception module with dimension reductions

Going deeper with convolutions. Szegedy, et al. *Proc. CVPR* 2015
Rethinking the inception architecture for computer vision. Szegedy et al. *Proc. CVPR* 2016
Inception-v4, Inception Resnet and the impact of residual connections on learning. Szegedy et al. Proc. AAAI conf. on AI, 2017.

ResNet-50

Problème

Augmenter la profondeur dégrade les performances du réseau

Pas lié à l'overfitting, mais à la taille du problème d'optimisation, avec grande diversité d'amplitudes de signal

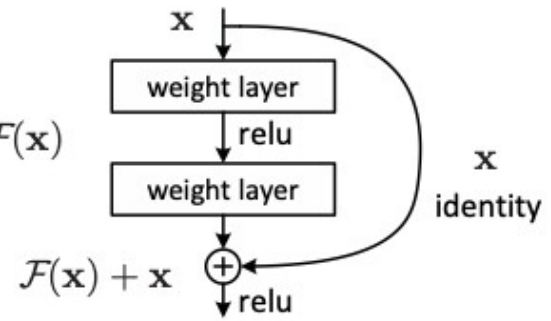


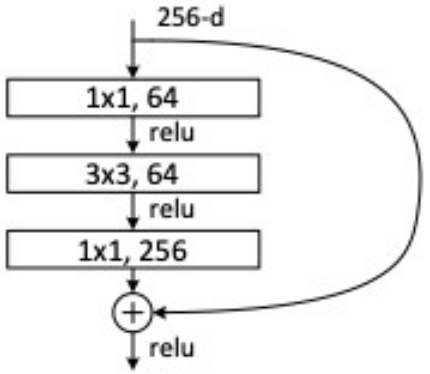
Figure 2. Residual learning: a building block.

Solution

Couches de convolutions modélisent le signal résiduel, d'amplitude plus petite : *shortcut connections*

Résultat

Top-5 error : 3.57 % (ILSVRC15, ensemble de 6 modèles)



Building block with projection for ResNet-50/101/152

Deep residual learning for image recognition. He et al. *Proc. CVPR,* 2016.

Bilan des CNN

Problème initial

- Pouvoir élargir les domaines d'influence (champ focal)
- Gros noyaux de convolutions : beaucoup de paramètres, et lents

Des solutions

- Convolutions à trous : pertes de détails, U-net (skip connections) meilleurs
- Convolutions 3x3 successives (VGG) : vanishing gradient → Inception puis ResNet

Tout va bien ?

- Difficile d'élargir le champ focal à toute l'image
- Les features éloignées n'influent que tardivement sur les features locales

REPRÉSENTATION...

Représentation (embedding)

Données brutes pas adaptées

Contiennent des informations non pertinentes (bruit)

Distances (e.g. MSE pour euclidienne) et autres fonctions de coût usuelles non adaptées

→ Espace (vectoriel) dans lequel on exprime les données via une transformation (projection)

DOG1

DOG2

BOAT

MSE(DOG1,DOG2)=150433 ; MSE(DOG1-BOAT)=74904 ; MSE(DOG2-BOAT)=113519

… ET CONTEXTE

Contexte

Une image est un ensemble d’éléments qui forment un contexte.

Analogie textuelle

Exemple de contexte : adjectif

C’était un avocat mûr.

J’ai vu un avocat compétent.

Voilà un bon avocat.

Affinement de la représentation

Token=mot

Représentation initiale commune pour le mot avocat

Correction = ajout d’un vecteur qui précise le sens (fruit, profession)

Illustration chaîne 3Blue1Brown (t=1:49 à 3:28): https://youtu.be/eMlx5fFNoYc?si=KTRcJtDXfCfB07DY&t=109

INTUITION DE L'ATTENTION

« C'était un avocat mûr » → entrées X_1 (C') à X_5 (mûr)

W_Q = « je suis nom commun en quête d'adjectif »

W_K = « je suis un adjectif »

W_V = « champ sémantique » (fruit, profession)

→ Q_1, ... Q_5 ; K_1, … K_5 ; V_1, … V_5

Produit scalaire $Q_i^T K_j$

=1 (ou valeur forte) si les deux sont vrais,

=0 (ou valeur faible) si non reliés

=-1 (ou valeur forte négative) si opposés

Résultat : valeur forte seulement pour $Q_4^T K_5$, V_5 → fruit

→ $$X_4 \leftarrow X_4 + \sum_j Q_4^T K_j V_j \approx X_5 + V_5$$

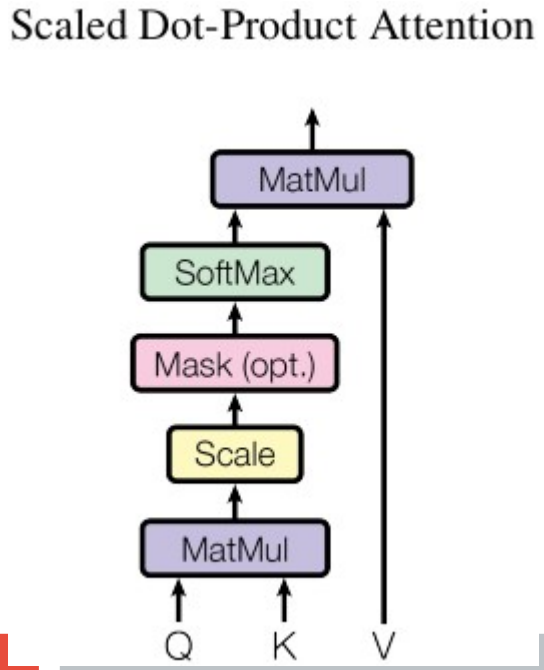
ATTENTION

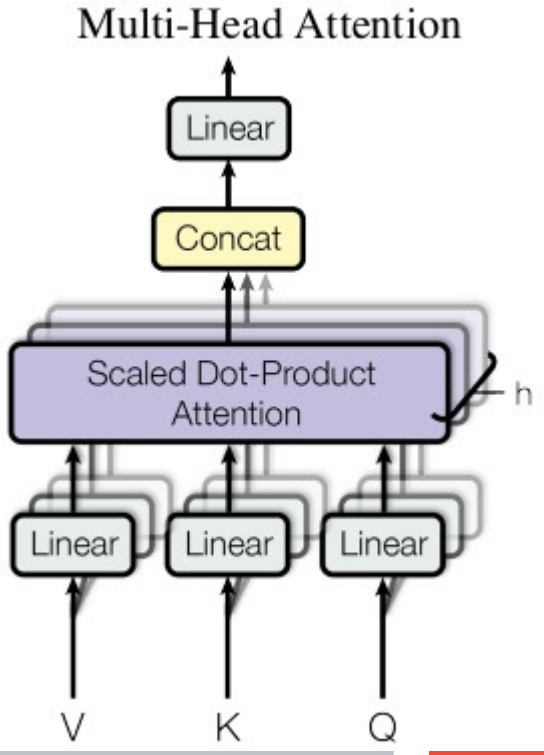
Notion d'attention

Requête Q, clé K, valeur V : projection (couche linéaire) du vecteur d'entrée X

Mesure de compatibilité Q vs K (produit scalaire) → carte de probabilité (SoftMax)

Pondération de V → vecteur d'ajustement





Source : Attention is all you need, Vaswani et al. NIPS2017

PositionAL encoding

La position est importante

« Cet avocat a mangé un avocat mûr »

→ l'ajustement vers « fruit » doit seulement s'appliquer au 2^{e} « avocat »

(note : « mangé » donnera une valeur de champ sémantique vers « être humain », lié à « profession »)

Positional encoding

Addition d'un code de position au vecteur de représentation

Manière triviale

On ajoute k, la position (entière) à toutes les composantes de l'embedding → problème d'échelle (embedding dans [-1,1], contexte ChatGPT=4096)

Normalisation selon la longueur de la séquence

Difficile de gérer des séquences variables

Constante est ajoutée à toutes les composantes

POSITIONAL ENCODING

PE sinusoïdal

Exemple fréquent :

mot en position k, d=taille représentation, indice i=0…d/2-1

$$p(k,2i)=\sin\left(\frac{k}{n^{2i/d}}\right); p(k,2i+1)=\cos\left(\frac{k}{n^{2i/d}}\right)$$

n=10000 dans *Attention is all you need*

Période x $n^{2/d}$ à chaque changement d'indice

Variations dans [-1,1]

Position relative (périodique)

Capture de la fréquence : tous les 2 mots, 3 mots… mais en continu

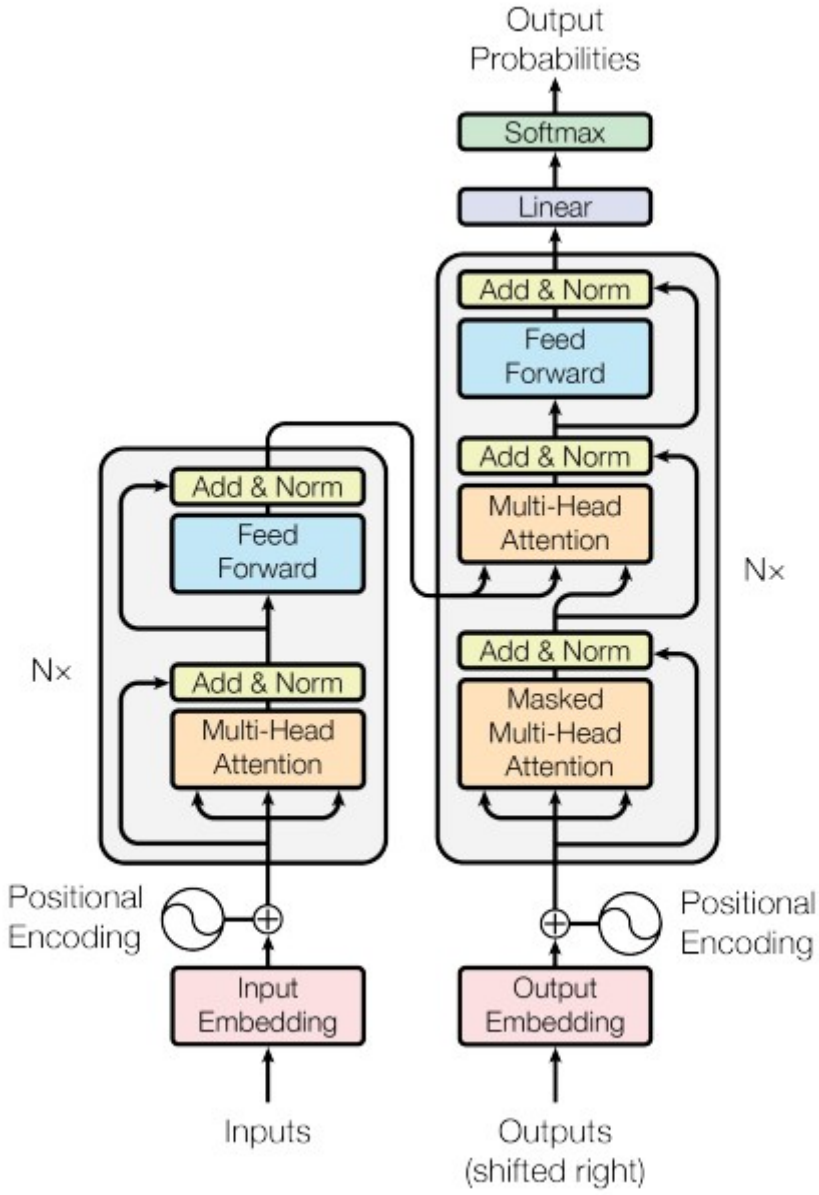
Distinction suivant index dans l'embedding

→ Peut être appris (e.g. ViT)

GÉNÉRATION DE TEXTE

Principe

- Découper la phrase en tokens (mots ou parties mots, ponctuation, etc)
- Entrée : une séquence de tokens
- Sortie : la séquence décalée (un mot en moins) avec un mot en plus (à trouver)
- Module d'attention : même nombre de vecteurs sortie qu'en entrée
- Transformation (encodage) de chaque token
  - Calcul des Q,K,V pour les tokens en entrée
  - Calcul des Q,K,V pour les tokens en sortie
  - Attention : Q de sortie avec les K en entrée → V en entrée
- Le mot suivant se déduit de l'encodage final du token lié



VISION TRANSFORMER

« An image is worth 16x16 words »

Redimensionnement de l'image en 224x224

Découpage de l'image en patches de 16x16 pixels

→ embedding de chaque patch dans un vecteur de dimensions 768 (projection linéaire)

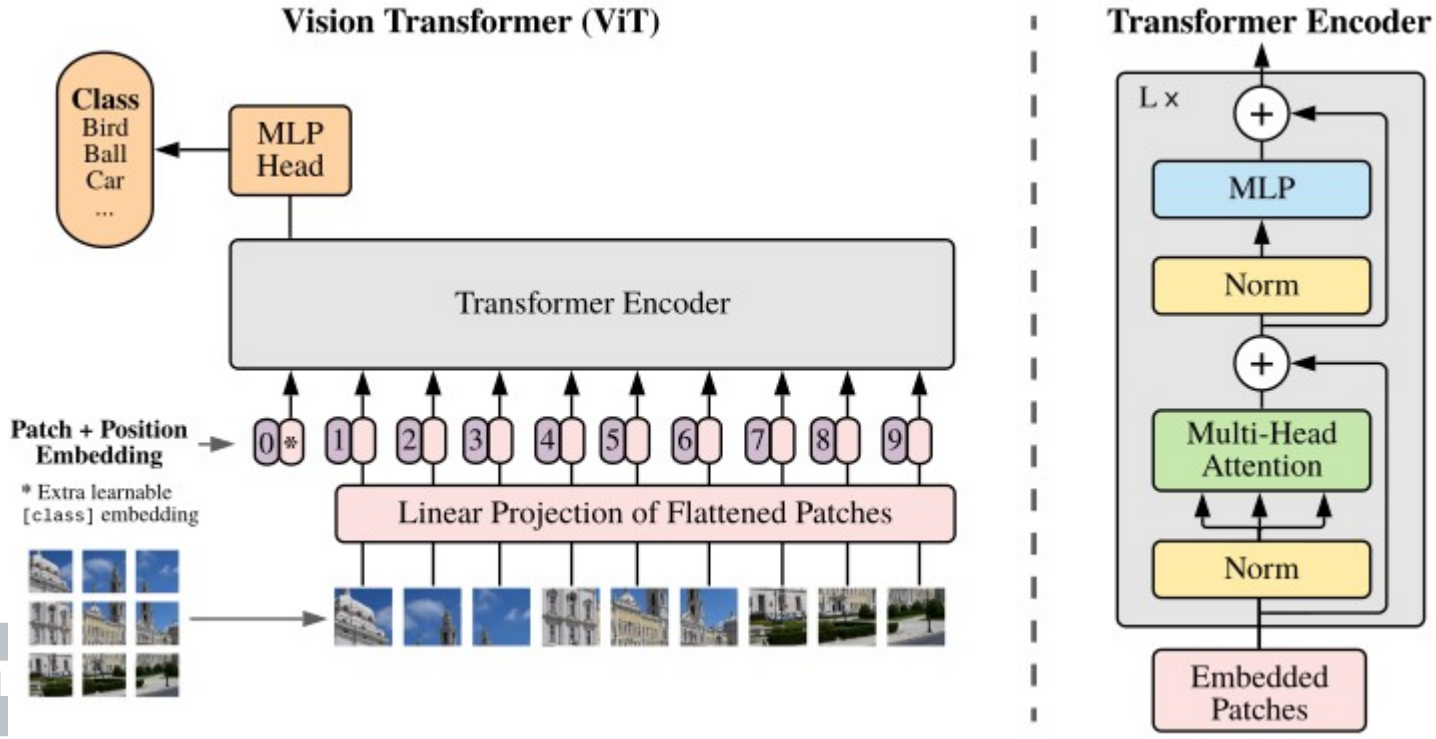
→ représentation de l'image comme 196(=14x14) mots de longueur 768 (14=224/16)

→ ajout d'un encodage positionnel (appris)

Couches d'encodage par attention multi-têtes

Ajout d'un token pour la représentation de la classe

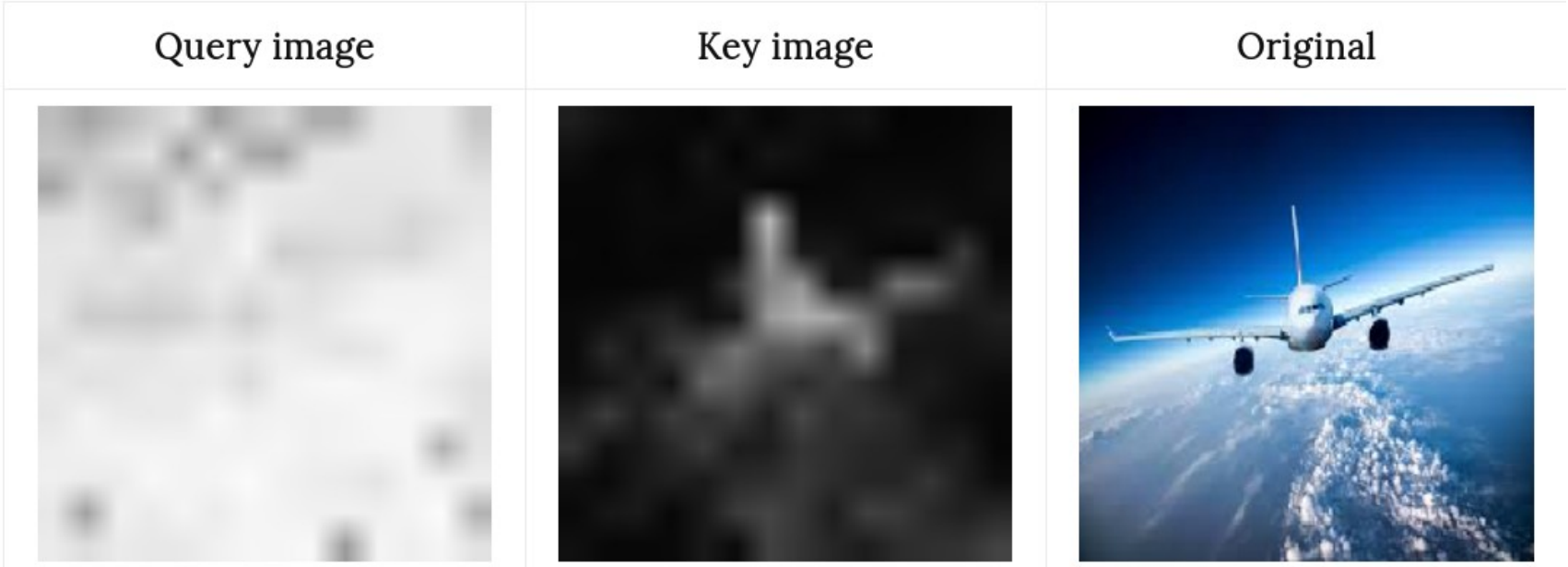
Encodage de la classe en sortie → MLP → vecteur de probabilité de classe



AUTRE VUE SUR Q,K,V

Analyse par composante (canal)

197-1 vecteurs → 196 valeurs par canal → on peut reformer une image basse résolution



Interprétation : j'ai trouvé un avion (Key), je veux en informer toute l'image (Query)

Source : J. Gildenblat Exploring Explainability for Vision Transformers

Tokens-to-Token ViT (T2T ViT)

Problème ViT

- Excellentes performances… mais training dataset énorme
- Petit dataset : séparation en token trop brutale → défaut sur détails locaux

T2T

- Module de calcul des embeddings des tokens plus évolué
- 3 étapes
  - Soft split
  - Token transformer
  - Reconstruct image

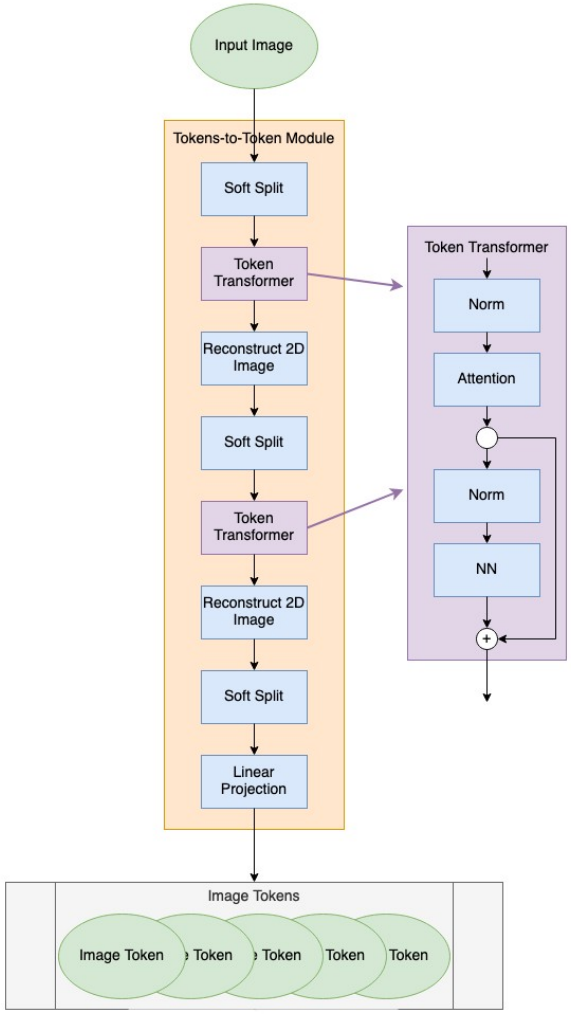


Image extraite de https://towardsdatascience.com/tokens-to-token-vision-transformers-explained-2fa4e2002daa/

Soft split

Les patches se chevauchent → augmentation du nombre de tokens

Token Transformer

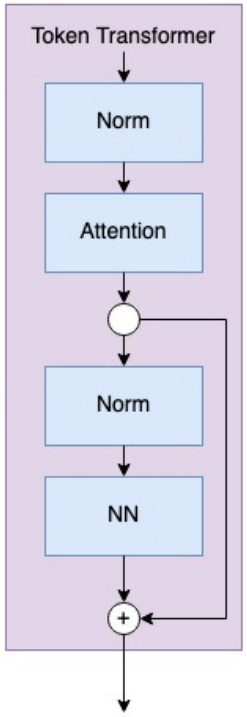
Même que bloc ViT

T’=MLP(MSA(T))

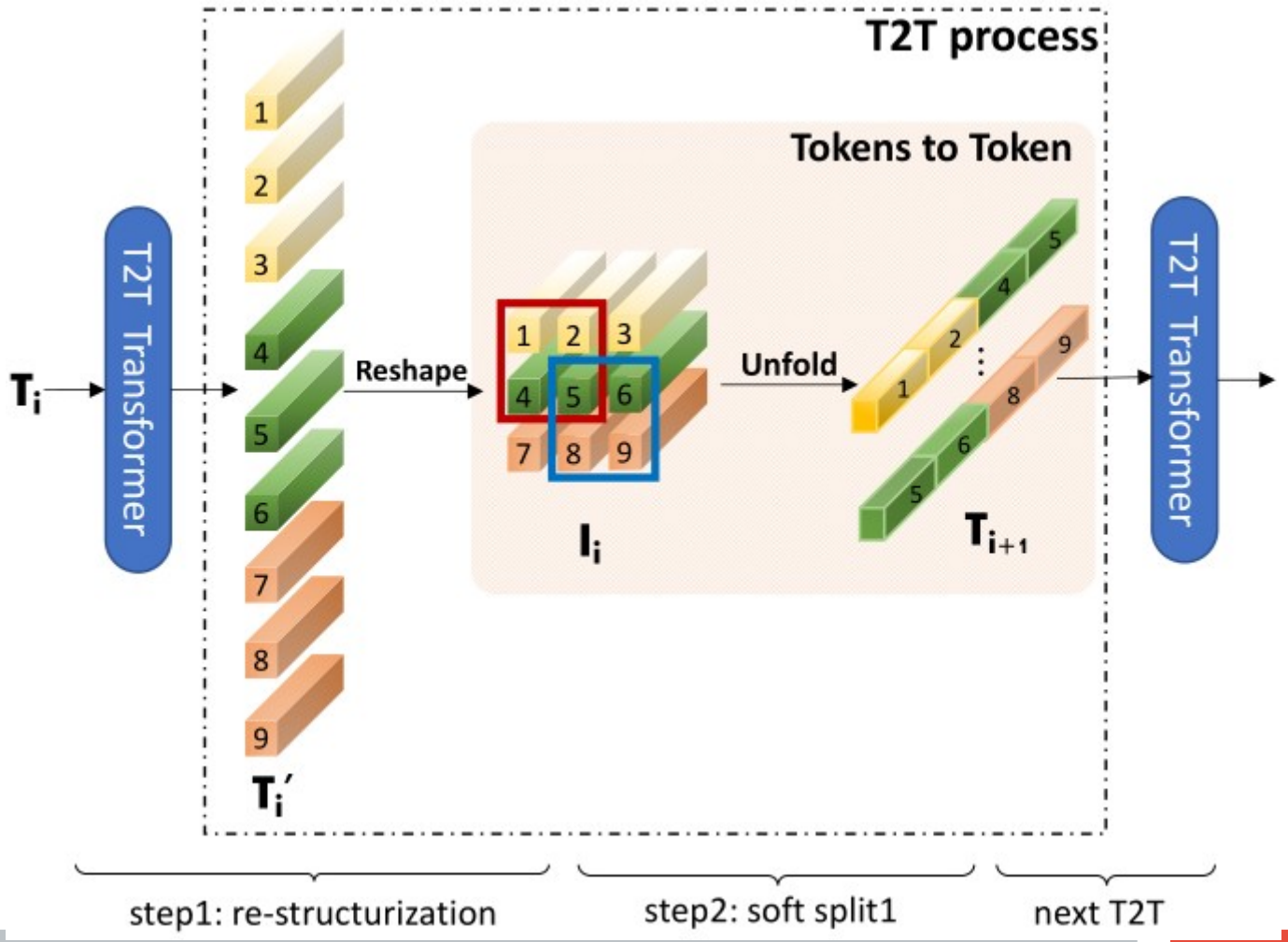
+ LayerNorm

+ MLP résiduel

T=patch linéarisé (ravel() en python)



RE-STRUCTURIZATION (RESHAPE)



Restructurization (Reshape

Soft Split 1

1,2,3,4,5,6,7,8,9,10,11,12,13,14

Soft Split 2

(1,2,8,9),(2,3,9,10),(3,4,10,11),

(4,5,11,12),(5,6,12,13),(6,7,13,14)

A RETENIR

ViT vs T2T-ViT : Complémentaires

- ViT : Classification, détection objets, reconnaissance d'objets, segmentation
- T2T-ViT : segmentation, génération, transfert de style, super-résolution

Traitement d'images

- Classification (ex reconnaissance d'objets, segmentation) ou régression (ex localisation d'objets, taille d'objets)
- Deux étapes :
  - Transformation des données dans un espace de représentation adapté
  - Techniques plus ou moins classiques de traitement de ces données vectorielles : notions d'estimateur et d'optimisation
- Aujourd'hui, ces deux étapes sont le plus souvent réalisées par des réseaux de neurones
- Mais des techniques plus classiques sont
  - Parfois plus efficaces
  - Bien utiles quand on a peu de données

RÉFÉRENCES

[Chen2017] L-C Chen et al. « Rethinking atrous convolution for semantic image segmentation » CoRR, 2017.

Playlist 3Blue1Brown (youtube) : en particulier, leçons DL5 et DL6

[Vaswami2017] A. Vaswami et al. « Attention is all you need ». NIPS 2017

[Dosovitskiy2021] A. Dosovitskiy et al. « An image is worth 16x16 words : Transformers for image recognition at scale ». ICLR 2021

Visualisation (site web) : J. Gildenblat Exploring Explainability for Vision Transformers

[Yuan2021] L. Yuan et al. « Tokens-to-Token ViT: Training vision transformer from scratch on ImageNet ». ICCV, 2021.