Introduction au Deep LEarning

Réseaux convolutionnels et images

Olivier Schwander <olivier.schwander@sorbonne-universite.fr>

Master MIND
Sorbonne Université

2025-2026

Insuffisance du MLP

Image 224×224 pixels avec 3 canaux RGB

- Modèle linéaire : $224 \times 224 \times 3 = 150\,528$ neurones
- MLP avec une couchée cachée à 1000 nœuds : 150 millions

Conséquences

- Trop de paramètres $\rightarrow$ risque de surapprentissage
- Besoin de trop d'exemples pour apprendre toutes les possibilités
- Perte de structure spatiale : les pixels voisins sont traités comme indépendants
- Pas d'invariance aux translations : une translation $\rightarrow$ besoin de nouveaux exemples
- Pas mieux pour les rotations, etc

Invariances et équivariances

Invariance

- Par translation, rotation, etc
- La classe doit rester la même

Équivariance

- Toujours par translations, etc
- Changement dans l'entrée $\rightarrow$ même changement dans la sortie

Partage des poids

- Plusieurs positions de l'entrée
- Même poids dans la couche
- Meilleure généralisation

Couche de convolution

Définition

- Entrée $X \in \mathbb{R}^{H \times W \times C_{\text{in}}}$
- Sortie $Y \in \mathbb{R}^{H' \times W' \times C_{\text{out}}}$
- Noyau $K \in \mathbb{R}^{k \times k \times C_{\text{in}}}$

$$Y[i,j,C] = \sum_{c=1}^{C_{in}} \sum_{u=0}^{k-1} \sum_{v=0}^{k-1} X[is+u, j \cdot s+v, c] K[u,v,c] + b$$

Balayage et partage des poids

- Le même noyau balaye chaque sous-région de l'image d'entrée
- On apprend aussi bien avec des motifs à un endroit qu'à un autre

Filtres

Détection de motif élementaire

- Interprétation traitement du signal
- Sortie du filtre : valeur élevée si le motif est présent, valeur faible s'il est absent

[Au tableau : motifs de plus en plus complexes et de plus en plus sémantiques]

Canaux de sortie

Besoin de plusieurs détecteurs

- Un par motif élementaire
- Toutes les rotations d'un gradient par exemple

$$Y[i,j,C] = \sum_{c=1}^{C_{in}} \sum_{u=0}^{k-1} \sum_{v=0}^{k-1} X[is+u, j\cdot s+v, c] K[u,v,c] + b$$

Noyau

- Tenseur $k \times k \times$ nombre de canaux de sortie
- On somme sur tous les canaux d'entrée

Paramètres

Ce qu'on apprend

- Le noyau
- Nombre de paramètres $k \times k \times C_{in} \times C_{out} + C_{out}$
- Exemple : $3 \times 3 \times 3 \times 64 + 64 = 1792$

Autre interprétation

- Une couche dense
- Mais avec des des poids partagés, forcés à être identiques

Hyper-paramètres

Choix

- Taille du noyaux k
- Stride S : pas pour déplacer la fenêtre
- Padding P : gestion des bords
- Nombre de canaux de sortie C_{out}

Quoi changer ?

- Taille $k = 3$, suffisant en génral à cause du champ réceptif
- Stride $S = 0$, on gère ça mieux plus tard avec le pooling
- Padding $P = 0$, information sur les bords de mauvaise qualité, et de toute façon on réduit la taille dans la suite du réseau
- Nombre de canaux de sortie : oui

Pooling

Max pooling

$$Y[i,j] = \max_{u,v \in \text{fenêtre}} X[i \cdot s + u, j \cdot s + v]$$

Entrée 4×4

1	3	2	4
2	5	1	3
4	2	6	1
3	1	2	5

Sortie 2×2

5	5
4	6

Motivation

- Réduction de dimension
- Invariance aux petites translations

Architecture de base

Partie convolutive

- Conv $\rightarrow$ ReLU (éventuellement plusieurs fois)
- Max Pool
- Conv $\rightarrow$ ReLU (éventuellement plusieurs fois)
- Max Pool
- ...
- Conv $\rightarrow$ ReLU (éventuellement plusieurs fois)
- Max Pool

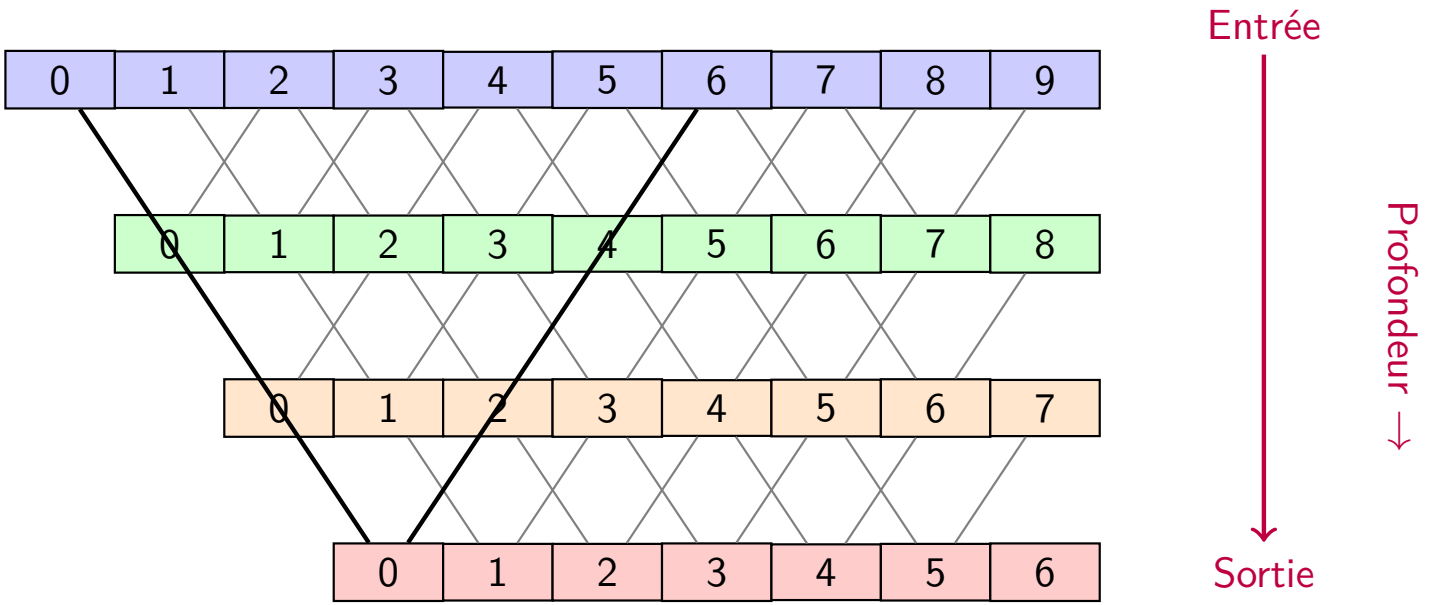
Partie dense

- Dense $\rightarrow$ ReLU
- ...
- Dense $\rightarrow$ ReLU
- Dense $\rightarrow$ Softmax

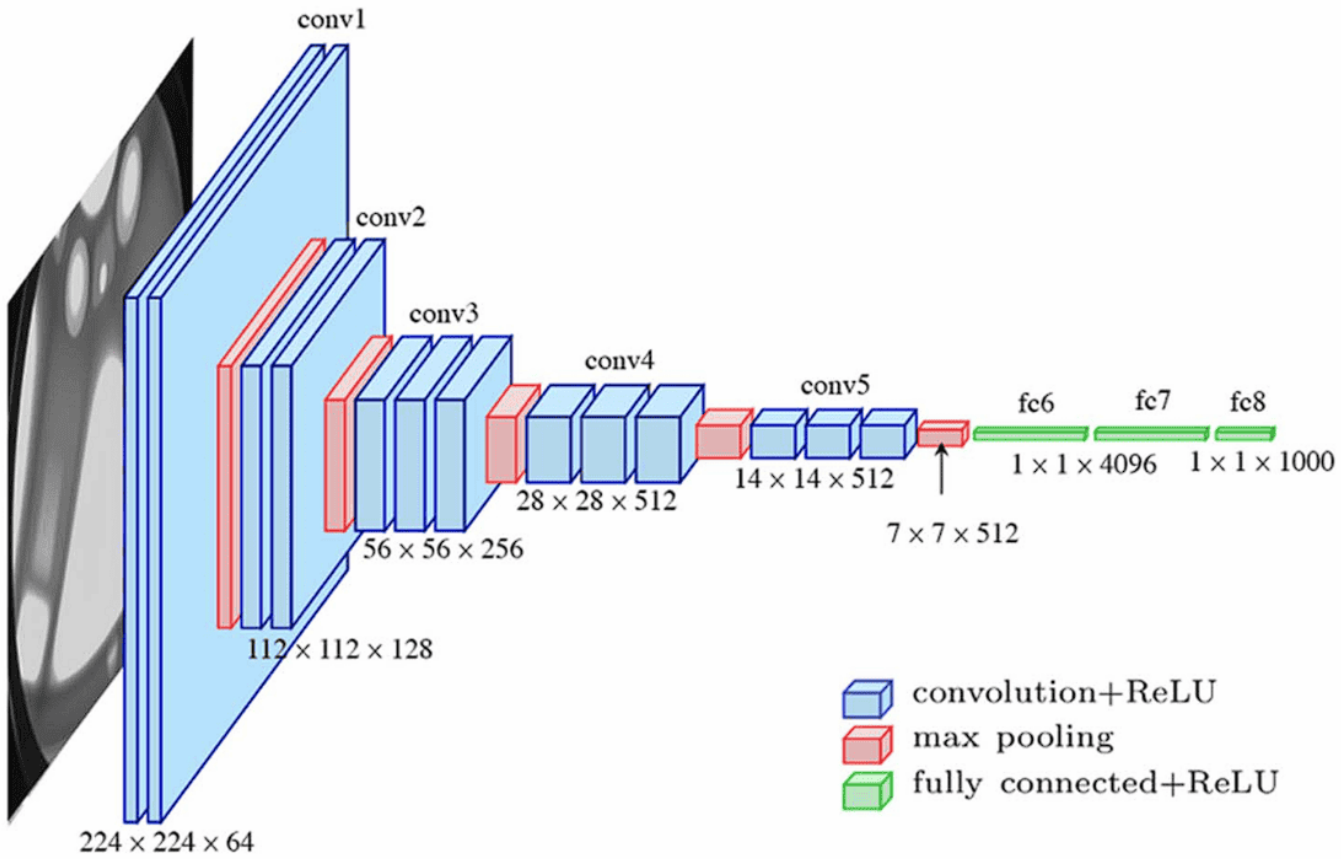
Champ réceptif

Définition

- Nombre de pixels d'entrée dont dépend une valeur en profondeur
- Un petit noyau avec de nombreux couches : grand champ réceptif



VGG



- VGG16 (138M parameters)
- Variantes : VGG11, 13, 16, 19

ResNet

Motivation

- *Degradation problem* : l'erreur de **train** augmente avec la profondeur
- Contre-intuitif : au pire il suffirait d'apprendre l'identité (ne rien faire dans les couches profondes)

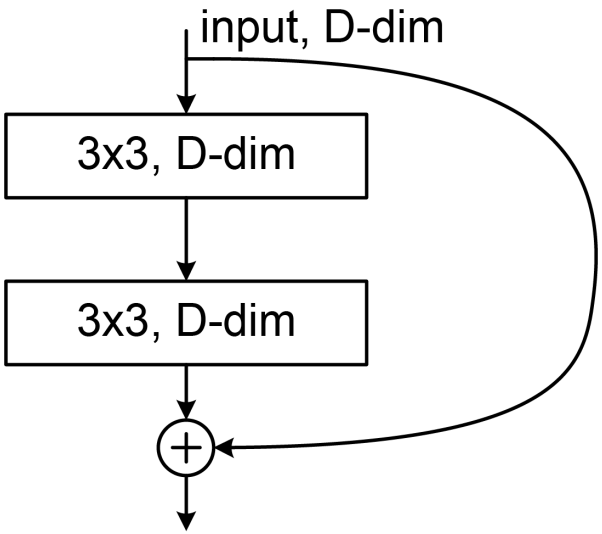
Couches résiduelles

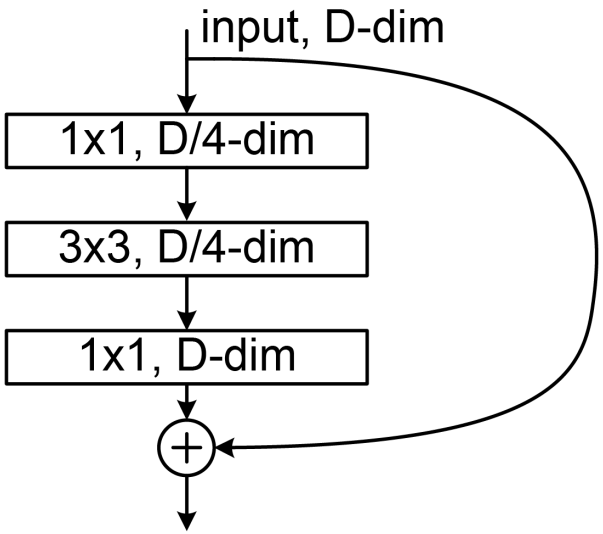
$$y = F(x) + x$$

- Facilite l'apprentissage de l'identité
- Réduit le *vanishing gradient* : dépendance additive et pas juste multiplicative

Couches résiduelles

Skip connections





Impact modernes

- LSTM et architectures récurrentes
- Dans les architectures *transformers*

Versions de ResNet

Vers les architectures vraiments profondes

Taille	Paramètres
ResNet-18	~11.7 M
ResNet-34	~21.8 M
ResNet-50	~25.6 M
ResNet-101	~44.5 M

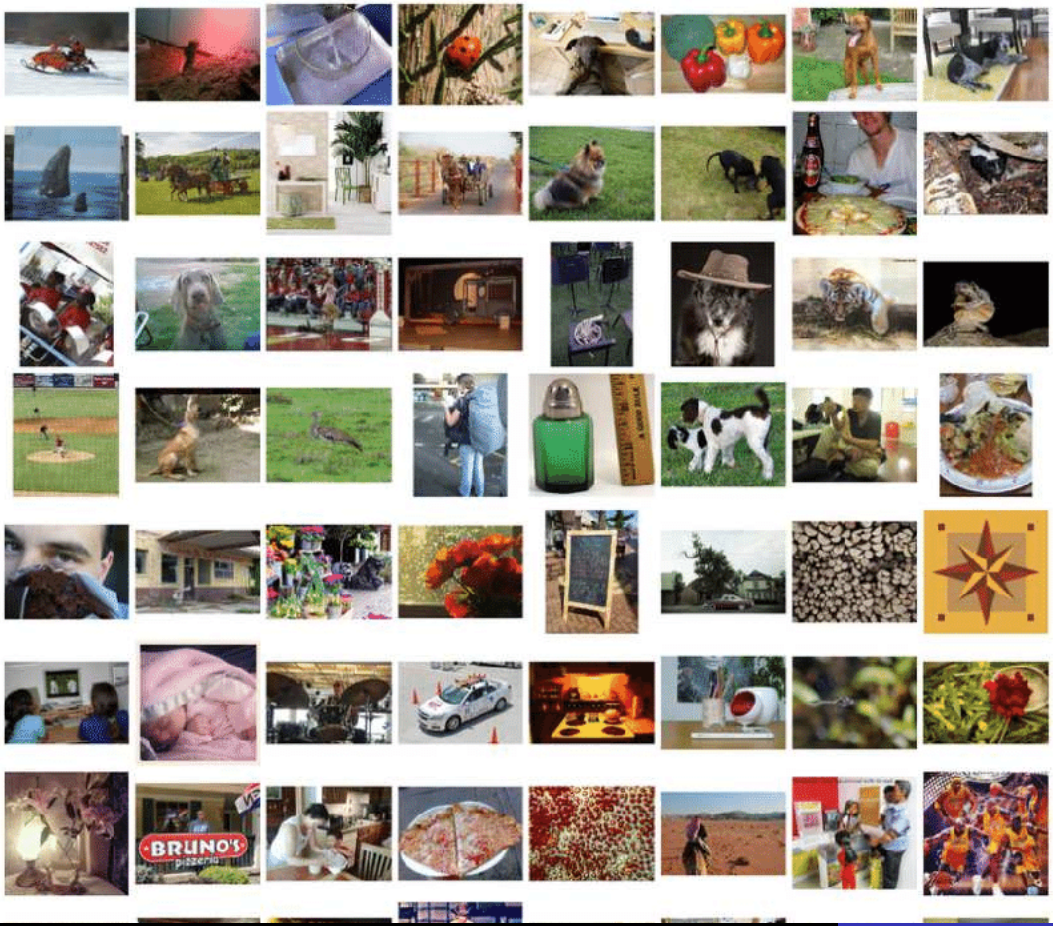
Nombre de paramètres

- Beaucoup moins que VGG, même avec beaucoup de couches
- Couches MLP final beaucoup plus petite

ImageNet

Dataset de référence

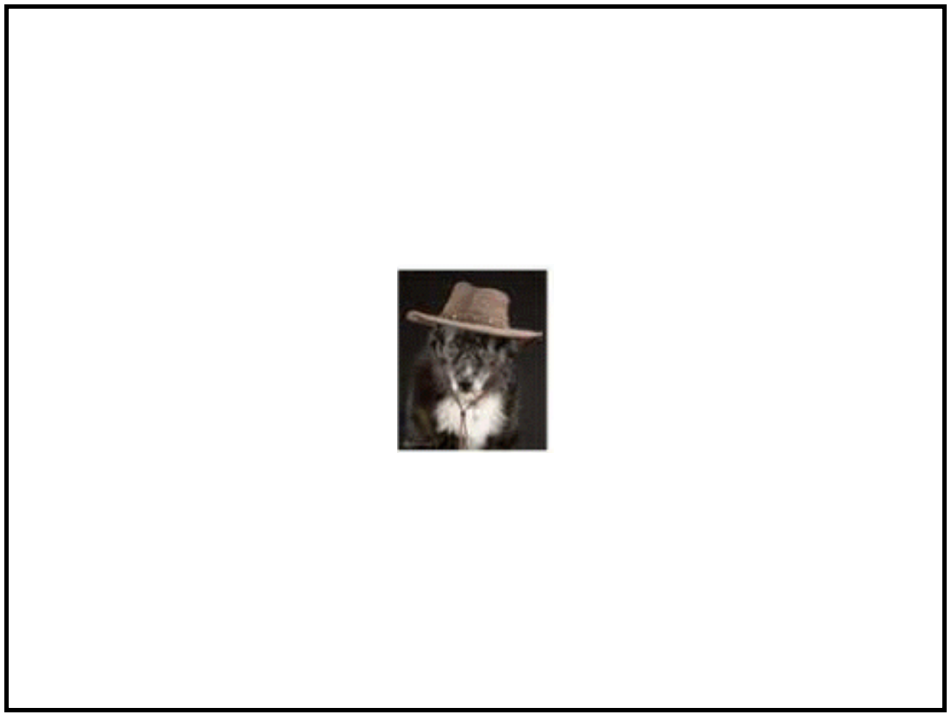
- 1M images, 1000 classes
- 224x224



Images plus grandes

Images plus réalistes

- Beaucoup plus que 224x224
- Sur un téléphone, 4000x3000 (sur le mien en tout cas)



Solution simpliste

Balayage manuel

- double boucle `for`
- On applique le classifier sur toutes les sous-régions possibles
- `OU` logique pour prendre la décision globale
- Très coûteux...
- (mais bonus gratuit : ça nous donne la posiition des objets d'intérêt)

Mais ?!

- Ce balayage, ça ressemble pas à une convolution ?

Réseau complètement convolutionnel

Taille arbitraire

- La seule chose qui bloque c'est le MLP final
- On le remplace par des convolutions

Astuce

- Noyau $1 \times 1 \times C_{in} \times C_{out}$

$$Y[i,j] = \sum_{c=1}^{C_{in}} X[i,j,c] \cdot K[0,0,c] + b, \quad K \in \mathbb{R}^{1\times 1\times C_{in}\times C_{out}}$$

Intérêt

- Kernel 1×1 préserve l'information spatiale
- Carte de prédiction en sortie, de taille variable
- Pooling global pour la décision finale

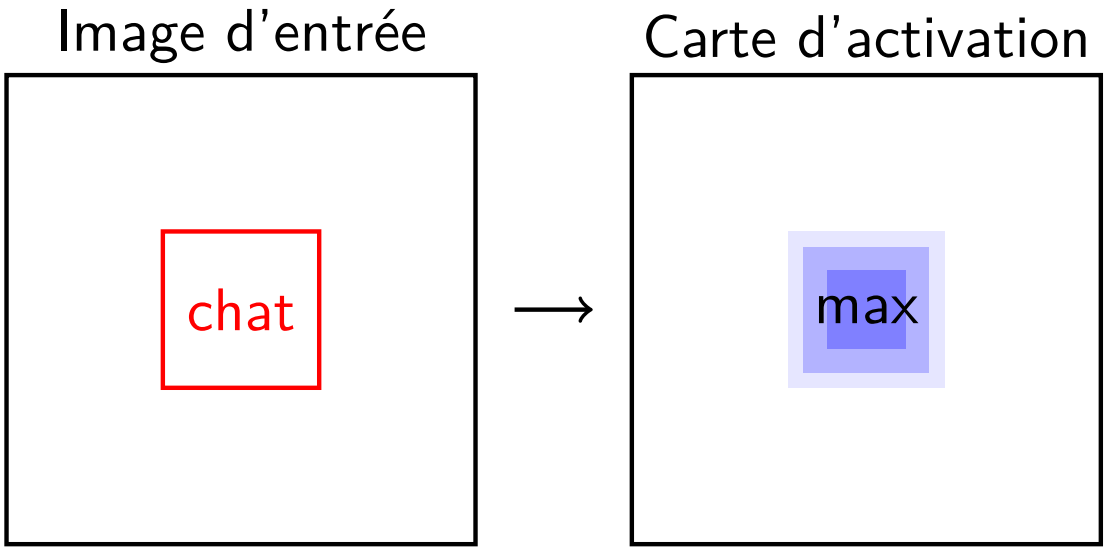
Localisation

Nouvelle tâche

- Trouver la position des objets
- Coordonnée, boîte englobante, voire même contour

Complètement convolutionnel

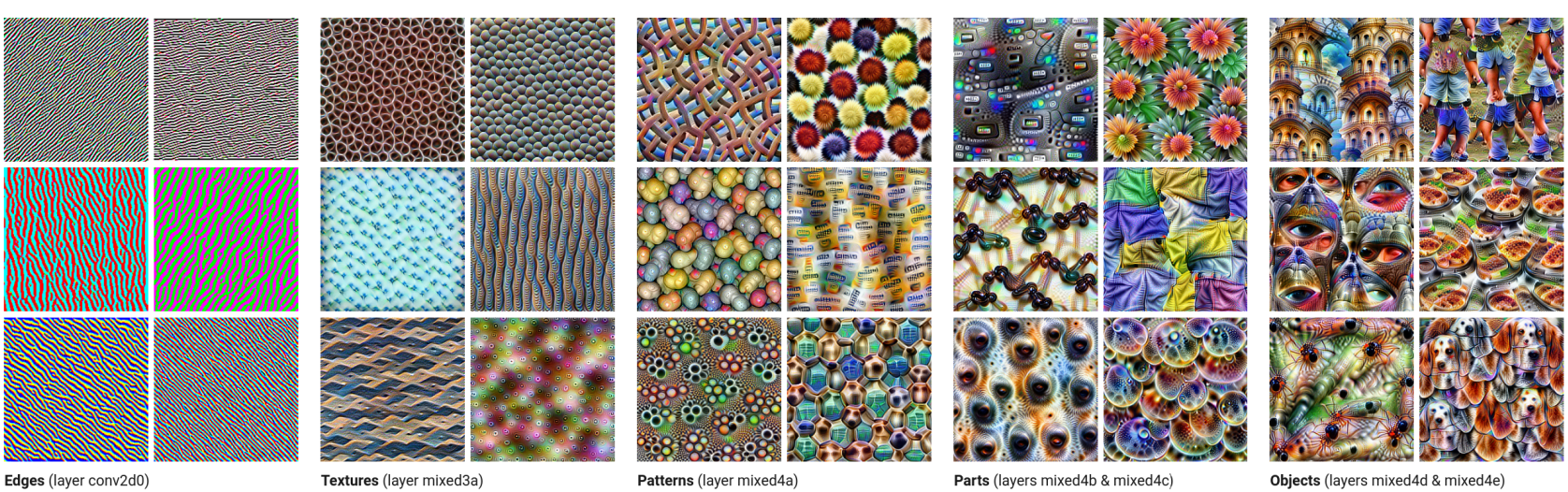
- Carte d'activation finale puis max
- Peu coûteux mais faible résolution



Analyse

Ce que voient vraiment les filtres

https://distill.pub/2017/feature-visualization/



- Vers l'entrée : features bas niveau
- Vers la sortie : features plus complexes, avec plus de sémantique

Les features bas niveau sont génériques (et en fait, pas que bas niveau)

Interprétation CNN

Deux parties

- **Représentation** : sorte de feature engineering, mais appris par rapport à la tâche
- **Classification** : des features tellement bonnes qu'un MLP suffit

$$\text{Image} \xrightarrow{\text{Couches Conv}} \text{Représentation} \xrightarrow{\text{Couches Dense}} \text{Classe}$$

Apprentissage end-to-end

- Apprentissage conjoint de la représentation et du classifieur
- À partir des images brutes, sans besoin de feature engineering

Modèles pré-entraînés

Sur ImageNet

- Excellentes performances en classification
- Poids disponibles sur internet (`torchvision.models`, HuggingFace, etc)

Point de départ

- Nouvelles tâches
- Nouvelles classes
- Nouvelles données

Modèles de fondation

- Réutilisables et adaptables

Fine-tuning

Tout réapprendre ?

- Pas assez de données
- Trop coûteux
- Pas nécessaire

Stratégie

- On change la tête dense du réseau
- On fixe tous les paramètres sauf cette nouvelle tête
- On apprend sur nos données

Justifications

- Features génériques et très bonnes
- Toujours mieux qu'une initialisation aléatoire

Fine-tuning en pratique

```
from torchvision.models import resnet50, ResNet50_Weights
model = resnet50(weights=ResNet50_Weights.DEFAULT)

# Nouveau format de sortie
model.fc = nn.Linear(model.fc.in_features, 10)

for param in model.parameters():
    param.requires_grad = False # Couches gelées
model.fc.requires_grad = True  # Seule la dernière couche

# Entraînement normal
for epoch in range(num_epochs):
    for images, labels in train_loader:
        ...
```

CLIP - Contrastive Language-Image Pre-training

Apprentissage de représenation

- Image encoder : CNN ou Vision Transformers
- Text encoder : Transformers
- Projection dans un espace latent commun
- Entraînement sur 400 millions de paires (image, texte)

Classification zero-shot

- Pas besoin de réentraînement
- Description textuelle des classes
- État de l'art sur ImageNet
- Modèle à utiliser en pratique de nos jours

VLM et chatbots

LLM + Images

- Entraînement multimodal

Exemples de tâches

- Visual Question Answering
- Captioning
- Localisation

Conclusion

Convolutions

- État de l'art pendant longtemps
- Base des modèles de fondation
- Base des stratégies modernes d'apprentissage

État de l'art moderne

- Maximiser le score $\rightarrow$ Vision Transformer (plus de CNN)
- Mais : moins de paramtères dans les CNN
- Encore intéressant pour des tâches simples
- Inférence peu coûteuse pour l'embarqué (temps, mémoire, énergie)