

GLO-4030/7030

APPRENTISSAGE PAR RÉSEAUX DE NEURONES PROFONDS

Plan de cours

Introduction

Fonctions d'activation

Plan de cours

Bienvenue à bord!

- 1ere édition
- L'équipe :



Mathieu
Carpentier



Alexandre
Gariépy

Laboratoires



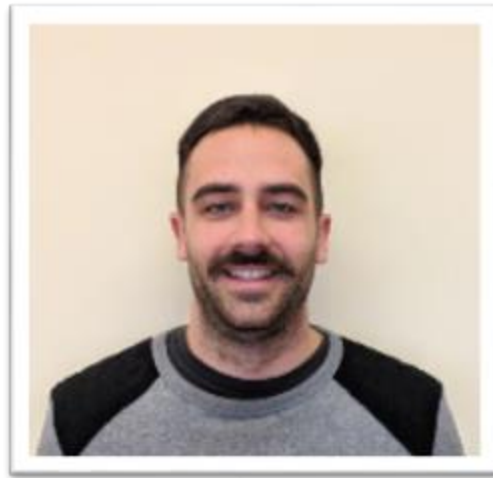
Ludovic
Trottier

Semaine #3
Optimisation

Cumul de 5 ans+ de Deep Learning

Bienvenue à bord!

- 2e édition
- L'équipe :



Nicolas
Garneau

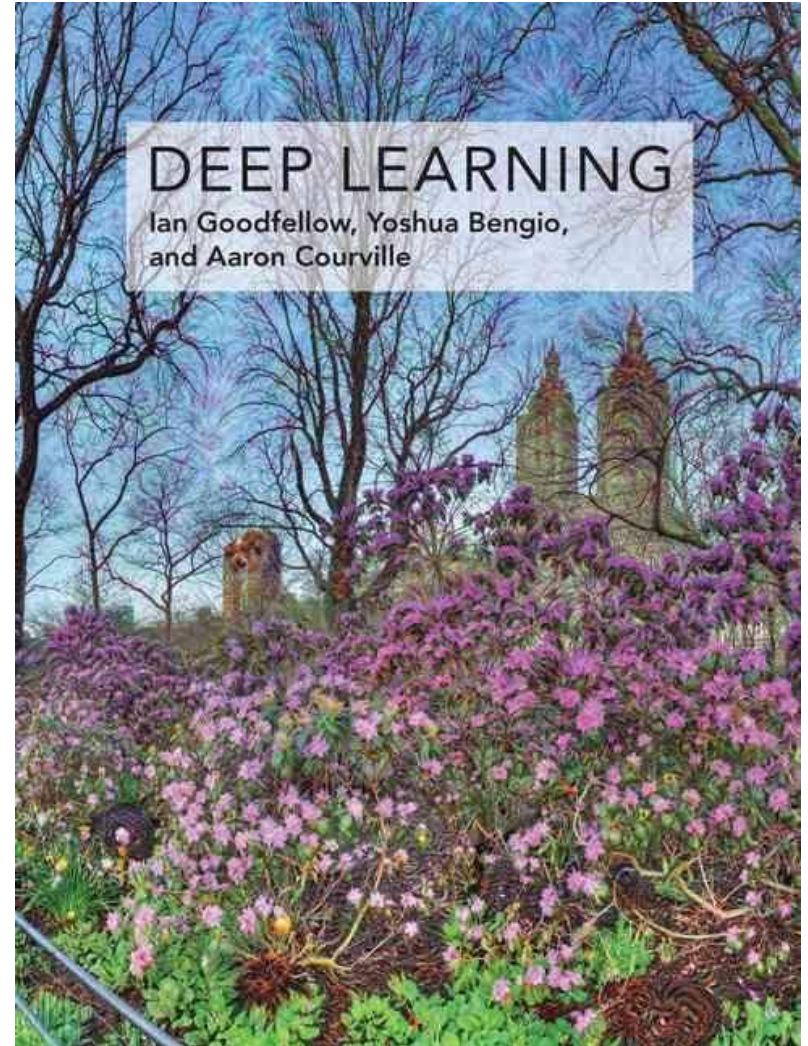
- Accès à des GPU via Jupyter notebook
 - NVIDIA K20
- <https://jupyter.calculquebec.ca/>
- Cœurs réservés pour les laboratoires :
réservation glo4030
- Gros merci à
 - Florent Parent
 - Félix-Antoine Fortin

Ressources GPU

- Google Cloud Platform
 - Envoyez-moi un courriel pour obtenir un crédit
- Amazon Web Services
 - Crédits étudiants + Github education

Manuel (obligatoire)

- Bonne référence
 - Version html gratuite
- <http://www.deeplearningbook.org/>



Pré-requis

- Python (laboratoire, TP)
- Certaine connaissance du machine learning
- Probabilité
- Algèbre linéaire
- Dérivée
- Un peu d'optimisation

Contenu du cours

- Un peu de théorie, oui...
- ... mais aussi de la pratique...
- ... et de la collection de timbres
 - nombre de techniques, trucs
 - 20+ exemples d'architecture
 - grand nombre de papiers
 - 30+ présentations orales
- Donner un aperçu du domaine, vous aider à démarrer dans la recherche



"All science is either
physics or stamp
collecting"
–E. Rutherford

Aperçu 1^{ère} moitié

- Connaissances de base (vertical)
 - Introduction, neurone, fondement apprentissage machine, fonctions d'activation
 - Graphes de calculs, fonctions de perte, rétro-propagation
 - Méthodes d'entraînement par optimisation, *batch norm*, initialisation des poids, trucs et astuces
 - Techniques de régularisation
 - Réseaux à convolution I
 - Réseaux à convolution II
 - Examen intra

Aperçu 2^{ème} moitié

- Concepts avancés (horizontal) :
 - Word embeddings
 - Autoencodeurs
 - Réseaux récurrents (RNN, LSTM et GRU)
 - Modèles d'attention, proposition de régions d'images, réseaux à mémoire
 - Apprentissage multitâches, pertes auxiliaires
 - Distillation (compression) des réseaux
 - Réseaux génératifs
 - Et +

Présentations affiches GLO-7030

- Sur votre travail de session
- Présentation style "poster session" en conférence
- Compte pour **12 %**

Librairie utilisée : **PYTORCH**

- Recommandation unanime des experts locaux (ils ont utilisés Theano, TensorFlow, Torch, Keras)
- Python, et non pas LUA
- Facile à débbugger
 - Vous pouvez extraire les données du GPU en tout temps
- Dérivation automatique autograd
- Support GPU pour Nddarray
- Package d'optimisation par descente de gradient inclus (SGD, RMSprop, ADAM, etc.)
- Beaucoup d'utilitaires (data loading, entraînement, data augmentation, torchvision etc.)
- Facile d'obtenir des modèles pré-entraînés

Examen

- Mi-Session
 - **35%** pour GLO-4030
 - **33%** pour GLO-7030
- Final
 - GLO-4030 seulement
 - Examen de 2 heures, **20 %**
- Pas de documents permis

Travaux pratiques

- 2 travaux
- Dans la première moitié du cours
- Total de **20 %**
- En python ou PyTorch

Projets

- Équipe de 1 à 2
- GLO-4030 : **25 %**
- Pour GLO-7030 : **35 %**
 - comme pas d'examen, projet devra être ambitieux (proche de publiable)
 - bonne méthodologie
- Trouvez un jeu de données proche de votre recherche / laboratoire / programme
- Vous pouvez utiliser langage et librairie de votre choix (TensorFlow, PyTorch, etc)
- Rapport sous format d'article scientifique

Sites web

- Site du cours :

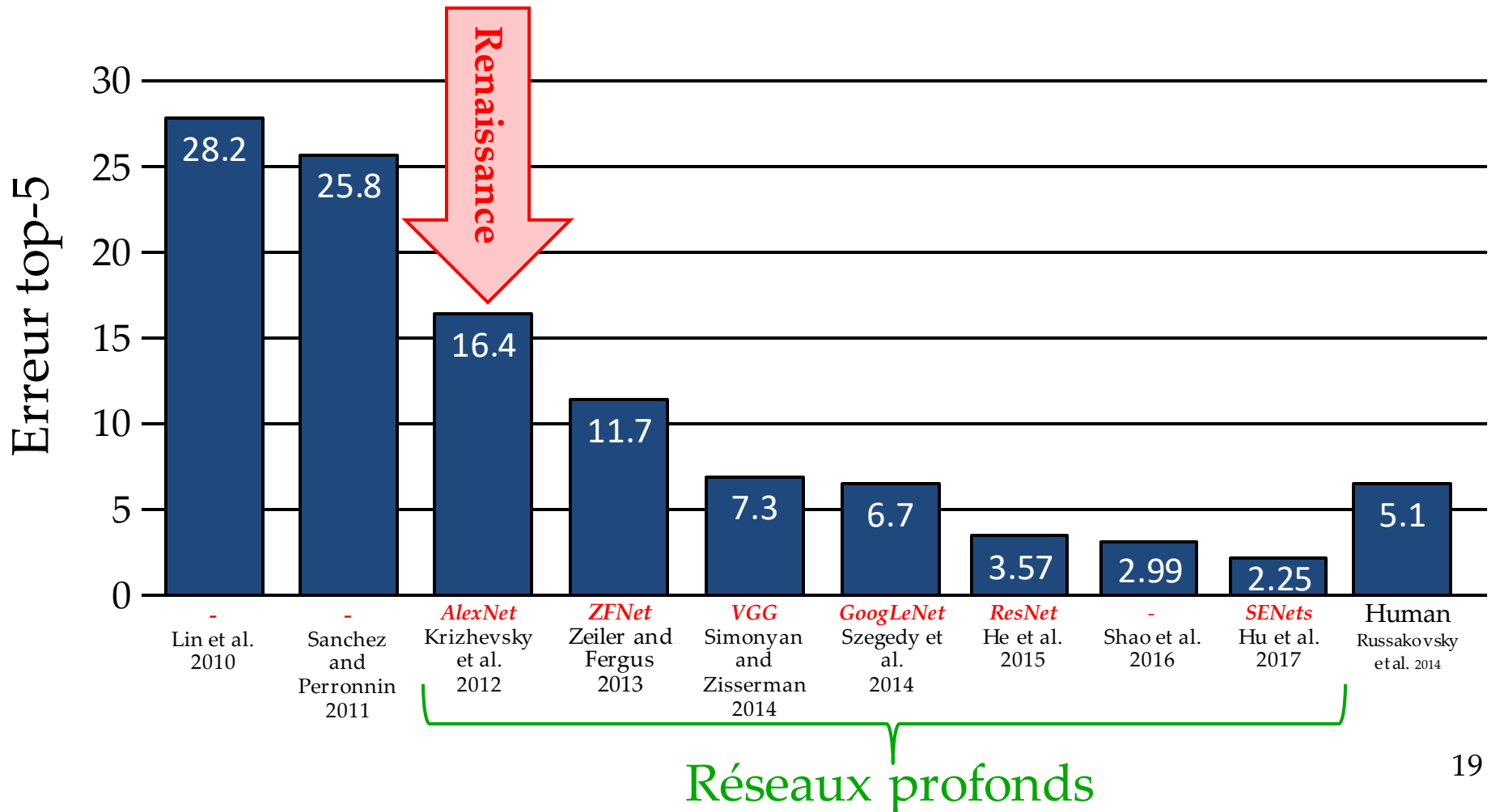
<https://ulaval-damas.github.io/glo4030/>

- Site pour le forum :
 - Forum MonPortail

Introduction

Large Scale Visual Recognition Challenge

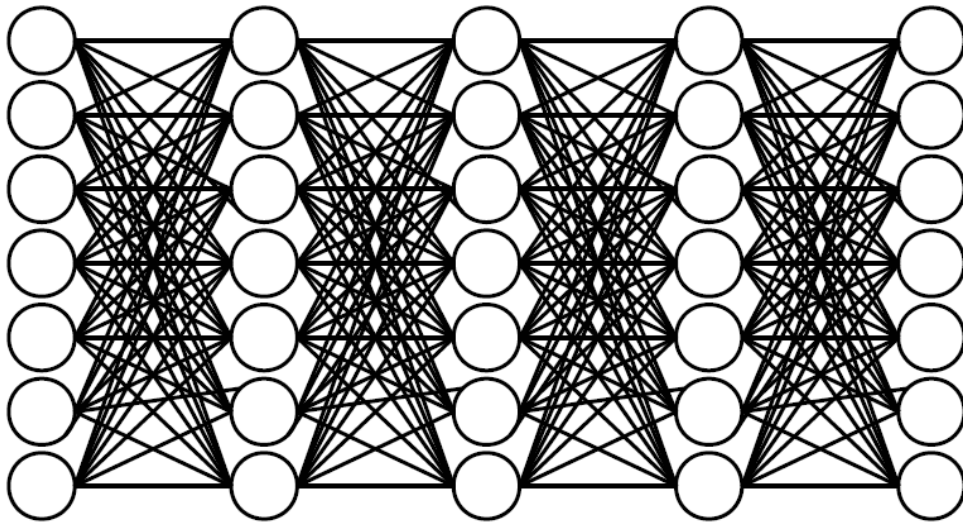
- *Image Classification Challenge* :
 - 1,000 classes d'objets
 - 1,431,167 images



Causes de la renaissance #1

Nouvelles fonctions d'activations

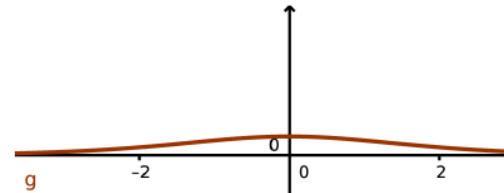
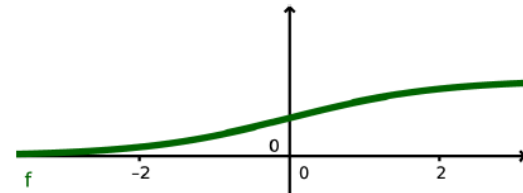
pré-
2006



$\nabla_{\theta} = 1e-15$ \leftarrow $\nabla_{\theta} = 1e-7$ \leftarrow $\nabla_{\theta} = 1e-3$ \leftarrow $\nabla_{\theta} = 0.1$ \leftarrow $\nabla_{\theta} = 10$

Vanishing gradient

Fonction logistique



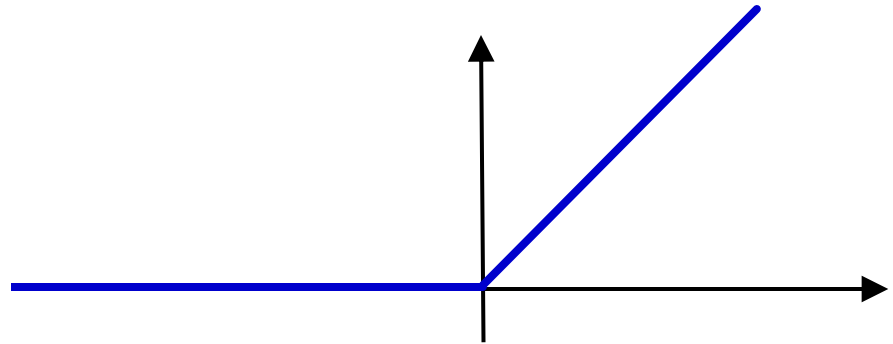
Sa dérivé

Causes de la renaissance #1

Nouvelles fonctions d'activations

- ReLU : Rectifier Linear Unit
- Introduite en 2010 par Nair et Hinton

$$f(x) = \begin{cases} x & \text{si } x \geq 0 \\ 0 & \text{sinon} \end{cases}$$



- Se calcule très rapidement : $\max(\text{input}, 0)$
- Beaucoup moins de *vanishing gradient*, car pente = 1 dans la partie active

- Grands jeux de données
- `www.image-net.org`
- 14 millions images, 22 000 catégories



Causes de la renaissance #3

Puissance de calcul via GPU



Novembre 2001



1,800 CAD

Rank	System	Cores	Rmax (GFlop/s)	Rpeak (GFlop/s)
1	ASCI White, SP Power3 375 MHz , IBM Lawrence Livermore National Laboratory United States	8,192	7,226.0	12,288.0
2	AlphaServer SC45, 1 GHz , HPE Pittsburgh Supercomputing Center United States	3,024	4,059.0	6,048.0
3	SP Power3 375 MHz 16 way , IBM DOE/SC/LBNL/NERSC United States	3,328	3,052.0	4,992.0
4	ASCI Red , Intel Sandia National Laboratories United States	9,632	2,379.0	3,207.0

12,000

Causes de la renaissance #3

Puissance de calcul via GPU



Juin 2005



2,995 USD

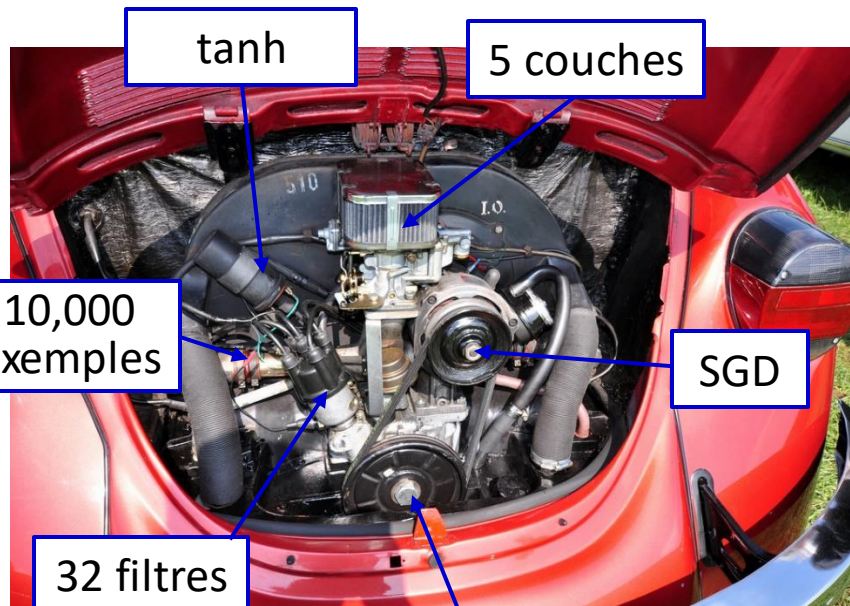
Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	BlueGene/L - eServer Blue Gene Solution , IBM DOE/NNSA/LLNL United States	65,536	136.8	183.5	716
2	BGW - eServer Blue Gene Solution , IBM IBM Thomas J. Watson Research Center United States	40,960	91.3	114.7	448
3	Columbia - SGI Altix 1.5 GHz, Voltaire Infiniband , HPE NASA/Ames Research Center/NAS United States	10,160	51.9	61.0	
4	Earth-Simulator , NEC Japan Agency for Marine-Earth Science and Technology Japan	5,120	35.9	41.0	3,200

110

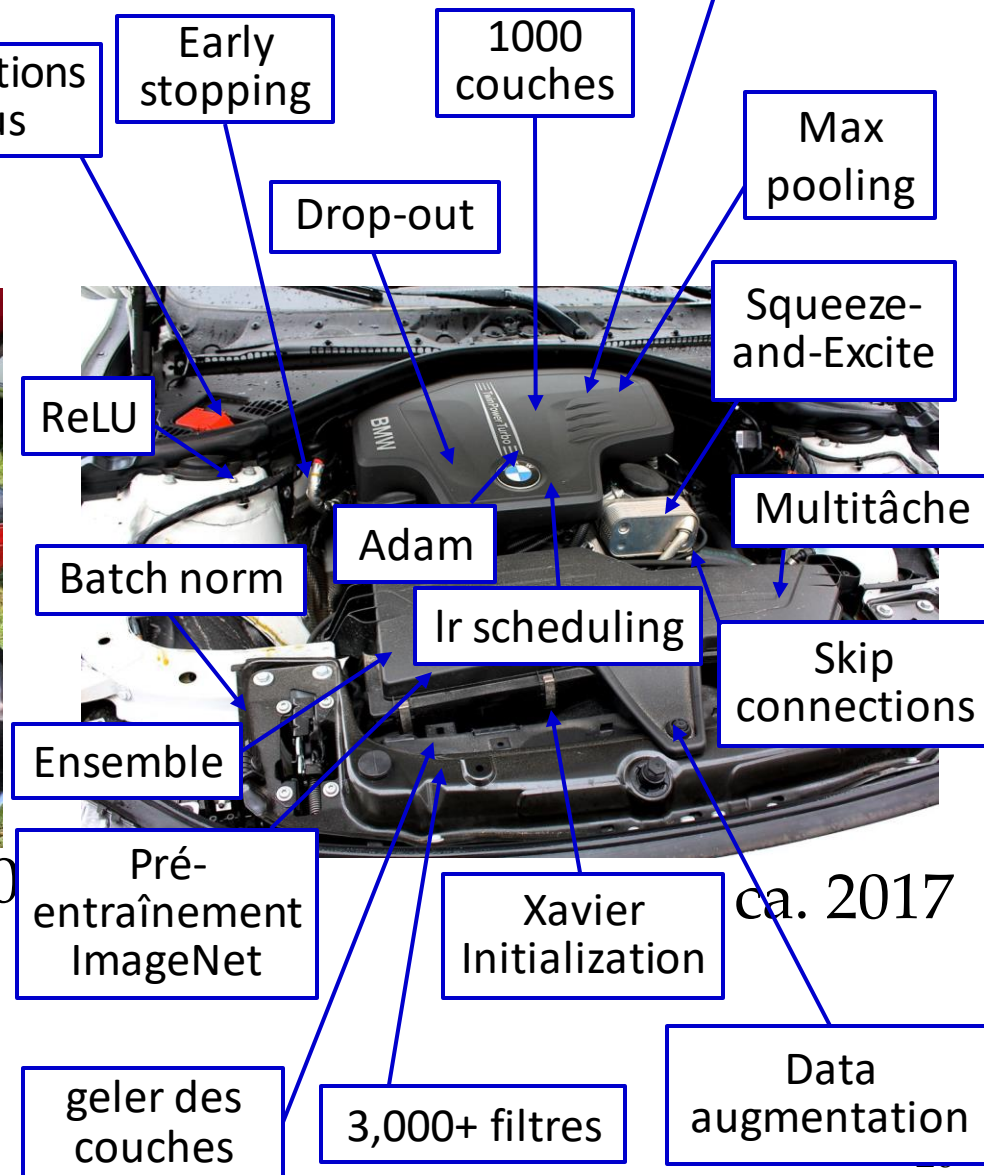
Continuation de la renaissance

- Progrès très rapide avec arXiv.org
- Plus d'une dizaine de soumission par jour
- L'article est souvent périmé lorsque présenté en conférence

Évolution des réseaux



ca. 1990



ca. 2017

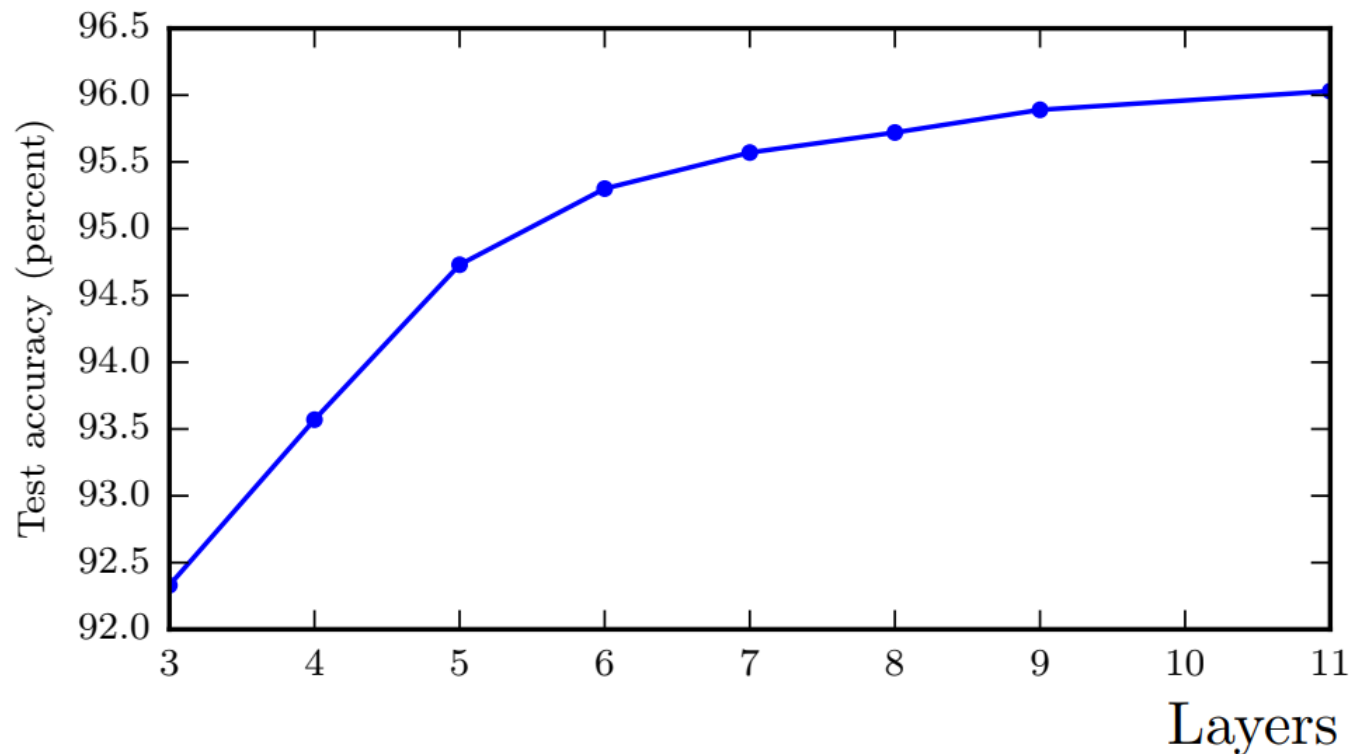
Pourquoi le Deep?

- Fonction mathématique extrêmement flexible et puissante (millions de paramètres)
- Théorème d'approximation universelle :
 - Peut approximer n'importe quelle fonction* avec un niveau de précision arbitraire
 - <http://neuralnetworksanddeeplearning.com/chap4.html>
- Réseaux peu profonds vont :
 - demander beaucoup de neurones (exponentiel)

*continue sur des sous-ensembles compacts de \mathbb{R}^n

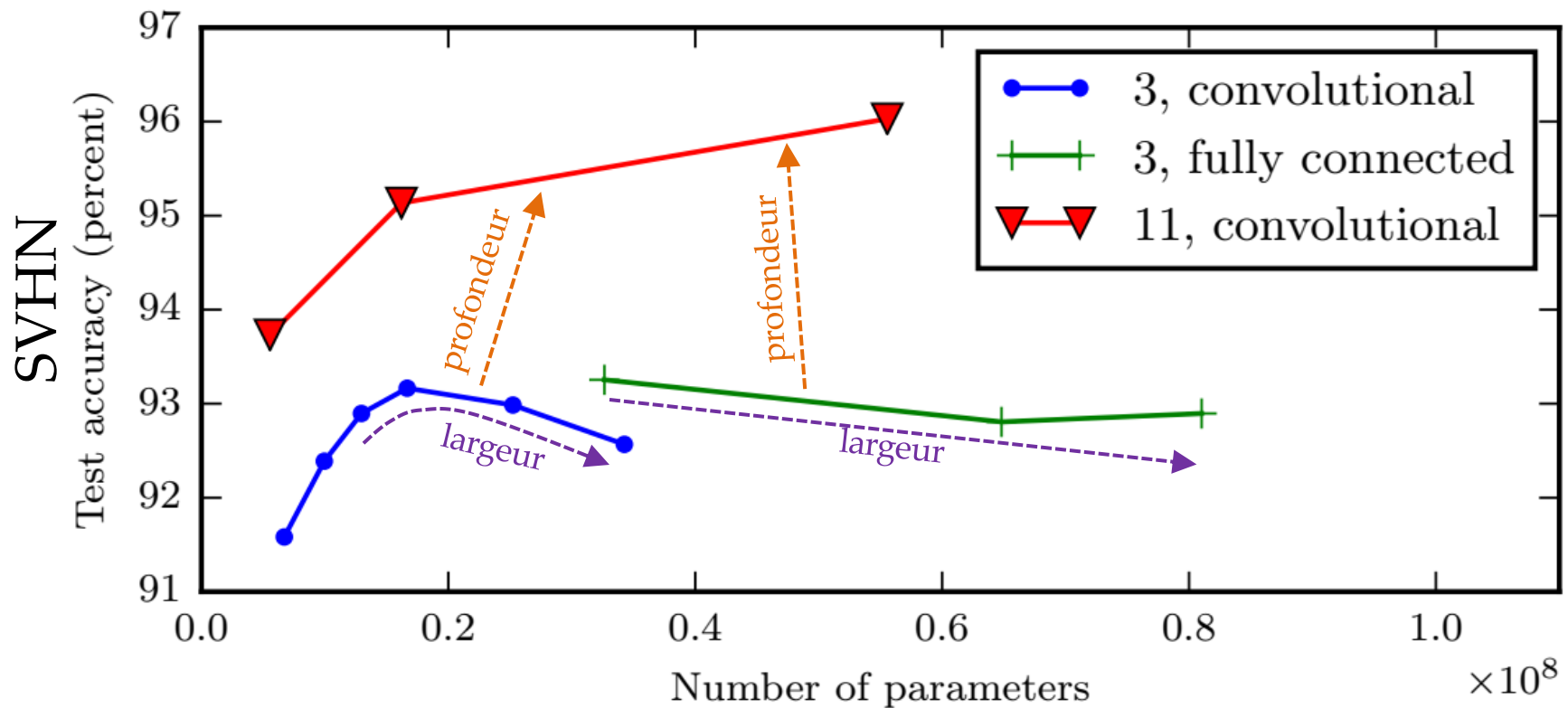
Généralisation vs. profondeur

- Street View Home Numbers SVHN

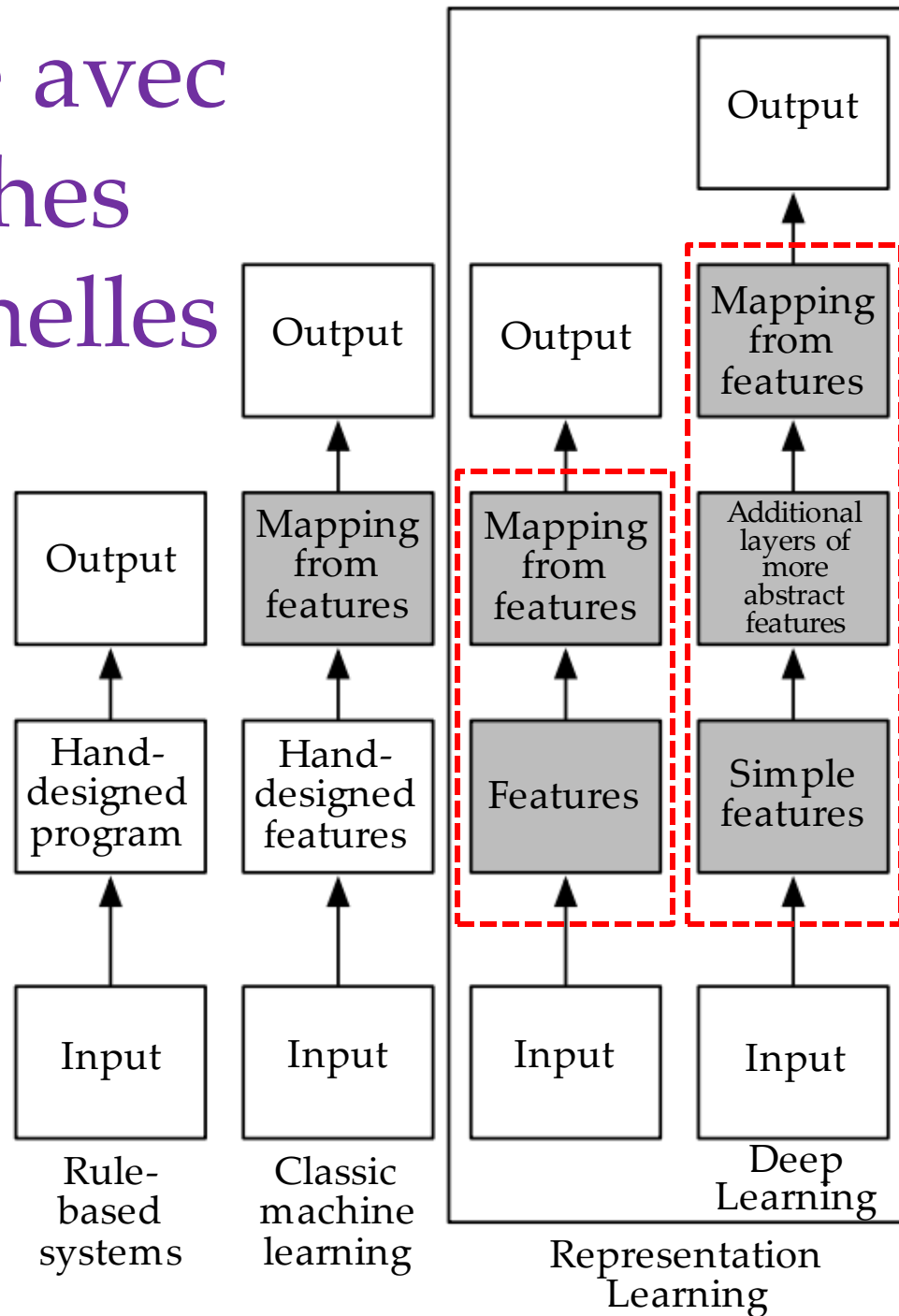


Meilleure généralisation

- Modèles larges ou peu profonds ont tendance à overfitter



Contraste avec approches traditionnelles



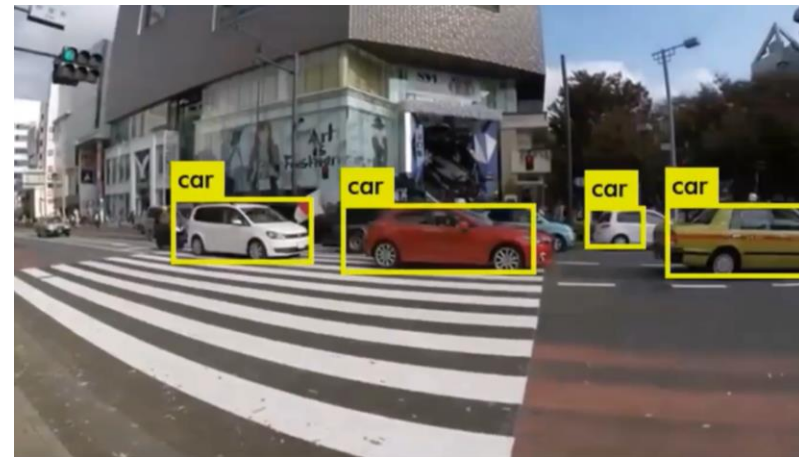
Appris
conjointement

Architecture = a priori

- Façon d'injecter de l'information a priori, via l'architecture
- Peut être vu comme un prior (parfois infiniment) fort
- Par exemple :
 - CNN (localité dans les images)
 - Maxpooling (invariance à la position)
 - RNN (dépendance temporelle)
 - Attention (régions plus informatives)
 - Spatial transformer network (déformation affines)
 - Softmax (appartenance à une seule classe)

Pourquoi le Deep en industrie ?

- Applicable à des solutions industrielles
- Entraînement (long) se fait en différé
- Tourne en temps réel sur machine GPU ou TPU (Tensor Processor Unit, Google)
- Le temps d'exécution dépend peu du nombre de classes
 - YOLO9000 : détection de **9000 objets**, 40 FPS
 - comment : un **seul pipeline commun** d'extraction des features



<https://www.youtube.com/watch?v=uG2UOasIx2I>

Pourquoi le Deep en industrie ?

- Si on découvre des nouveaux cas problématiques, on les ajoute dans la banque d'entraînement
 - facile à expliquer à un non-expert
- La quantité de données d'entraînement n'influe pas sur le temps d'inférence*
- Systèmes experts (*explicite*) deviennent fragiles avec nombre croissant de règles
 - réseaux (et ML) : *implicite*

*bien entendu, on pourrait augmenter la taille du réseau si on a beaucoup de données, afin d'en profiter

Deep Learning is eating software

The pattern is that there's an existing software project doing data processing using explicit programming logic, and the team charged with maintaining it find they can replace it with a deep-learning-based solution. I can only point to examples within Alphabet that we've made public, like upgrading search ranking, data center energy usage, language translation, and solving Go, but these aren't rare exceptions internally. What I see is that almost any data processing system with non-trivial logic can be improved significantly by applying modern machine learning.

This might sound less than dramatic when put in those terms, but it's a radical change in how we build software. Instead of writing and maintaining intricate, layered tangles of logic, **the developer has to become a teacher, a curator of training data and an analyst of results.** This is very, very different than the programming I was taught in school, but what gets me most excited is that it should be far more accessible than traditional coding, once the tooling catches up.

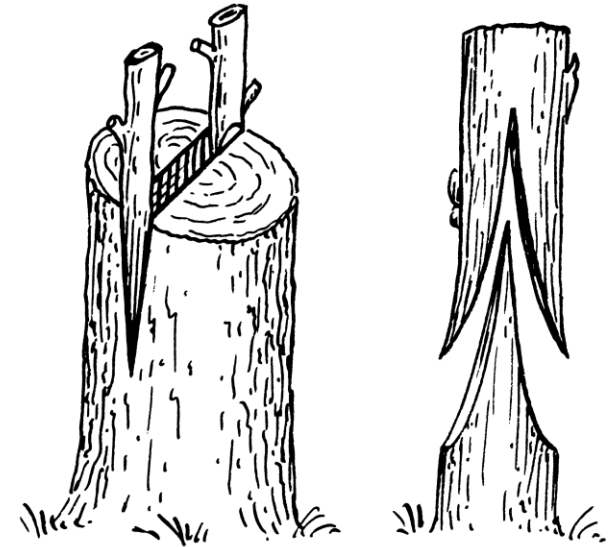
<https://petewarden.com/2017/11/13/deep-learning-is-eating-software/>

Transfert d'innovation

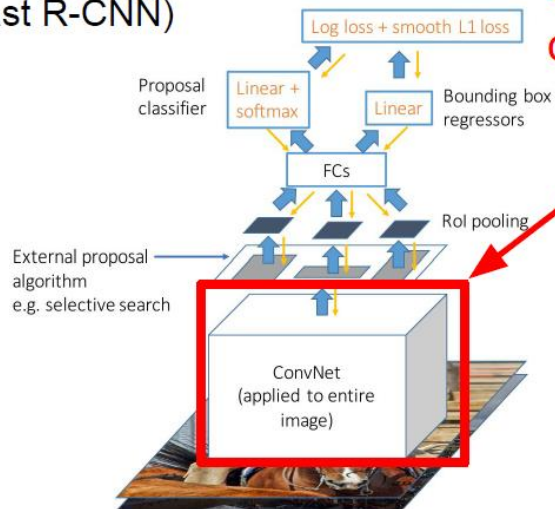
- Les innovations dans les architectures de réseaux faits pour une tâche X ont tendance à aussi aider pour de nombreuses autres tâches
- Synergies et emballements

Bouturage

- Combiner des réseaux pré-entraînés sur des tâches différentes

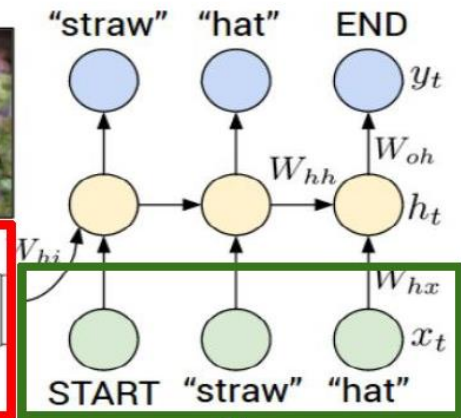
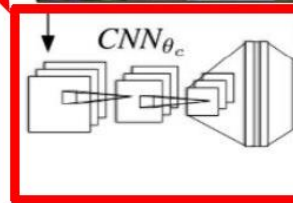


Object Detection
(Fast R-CNN)



CNN pretrained
on ImageNet

Image Captioning: CNN + RNN

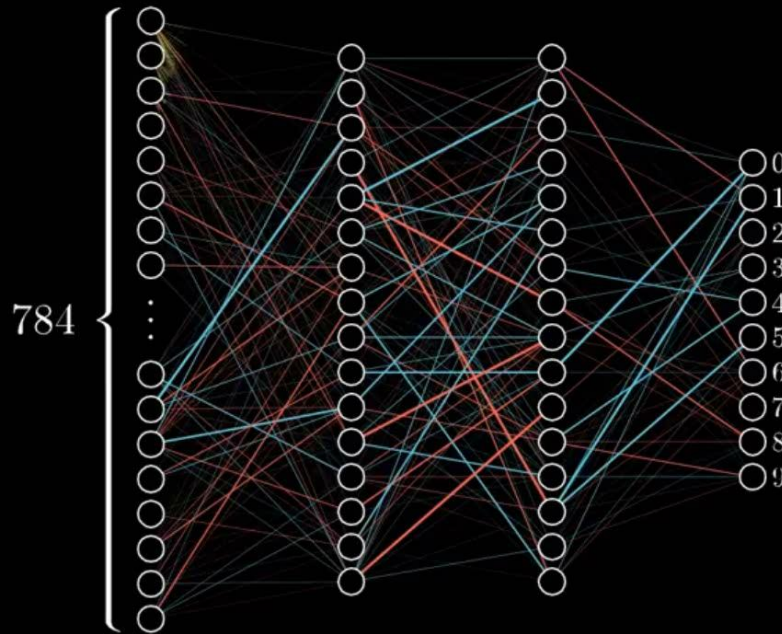


Word vectors pretrained
with word2vec

Le gradient = sève

- Pas de gradient = pas d'apprentissage

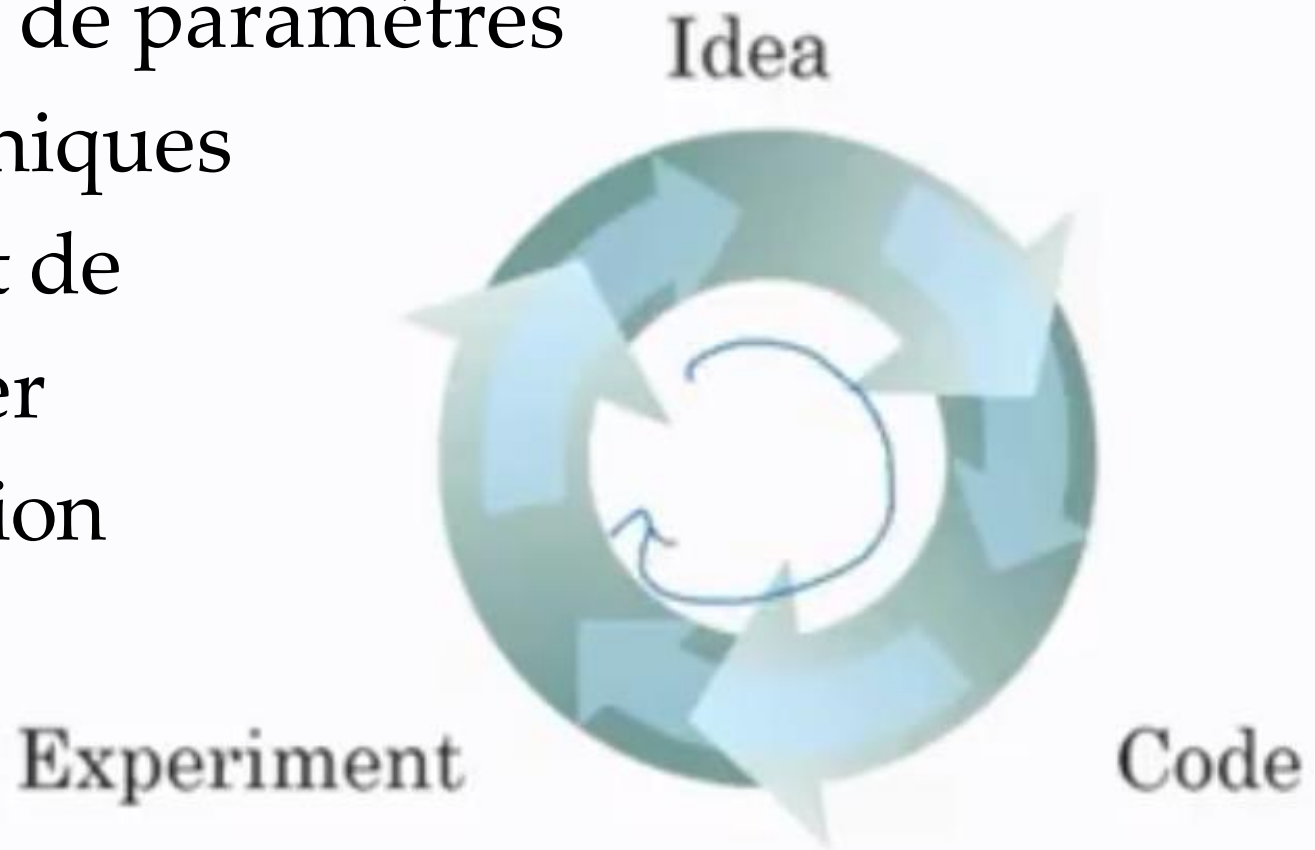
Training in progress...



Processus itératif

Les voies du réseaux sont impénétrables

- Beaucoup de paramètres et de techniques
- Important de développer une intuition



Crédit : Andrew Ng

Optimisation vs. gradient

- Théorème d'approximation universelle ne dit pas comment trouver cette fonction
- Relation incestueuse entre les architectures développées (dérivable *end-to-end*) et les méthodes d'optimisation
- Autres approches (Hebbian), mais on n'en parlera pas dans le cours

Toujours besoin de beaucoup de données?

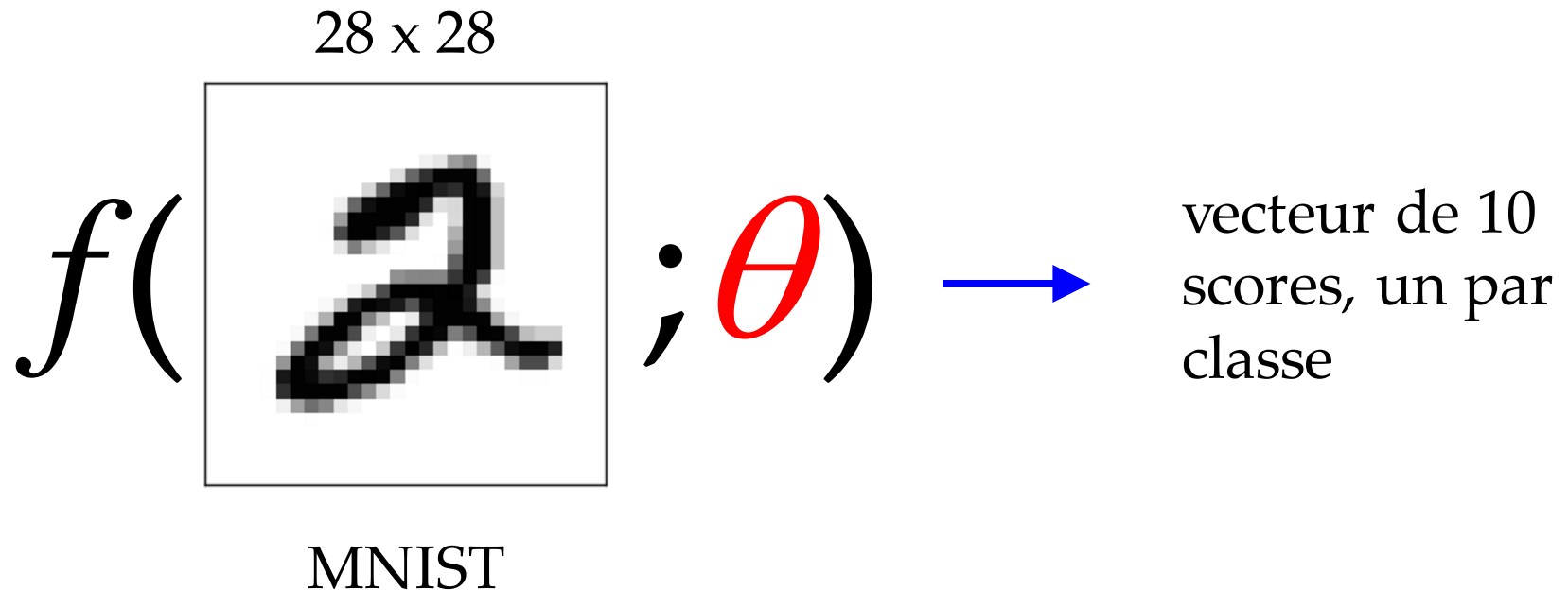
- Non, si l'on passe par des réseaux pré-entraînés
- Fantastiques extracteurs de caractéristiques
- Résultats dépassent souvent l'état de l'art pré-Deep

Deep Learning : appel à la prudence

- Excellente présentation NIPS 2017 d'Ali Rahimi
- <https://youtu.be/Qi1Yry33TQE>
- Deep Learning is alchemy
- Doit chercher plus de rigueur, comprendre ce qui se passe

Exemples d'applications

Reconnaissance de caractères



(Labo 1) θ : paramètres de la fonction

Reconnaissance d'images

224 x 224 x 3

$f($



$;\theta)$



vecteur de 1000
scores, un par
classe

ImageNet

θ : paramètres de la fonction

Note : les fonctions
seront différentes
d'un exemple à l'autre

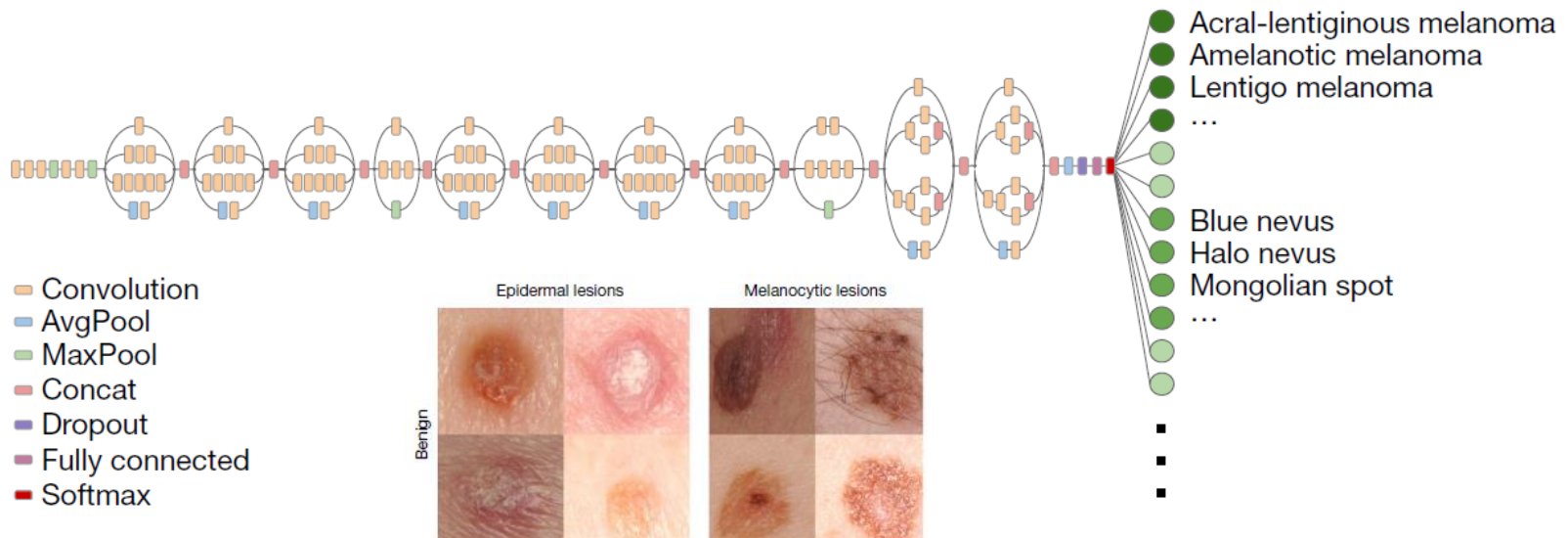
Dermatologist-level classification of skin cancer with deep neural networks

Andre Esteva^{1*}, Brett Kuprel^{1*}, Roberto A. Novoa^{2,3}, Justin Ko², Susan M. Swetter^{2,4}, Helen M. Blau⁵ & Sebastian Thrun⁶

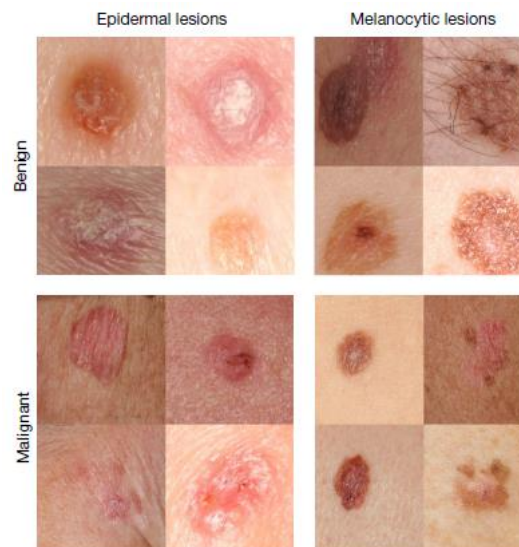
Skin lesion image

Deep convolutional neural network (Inception v3)

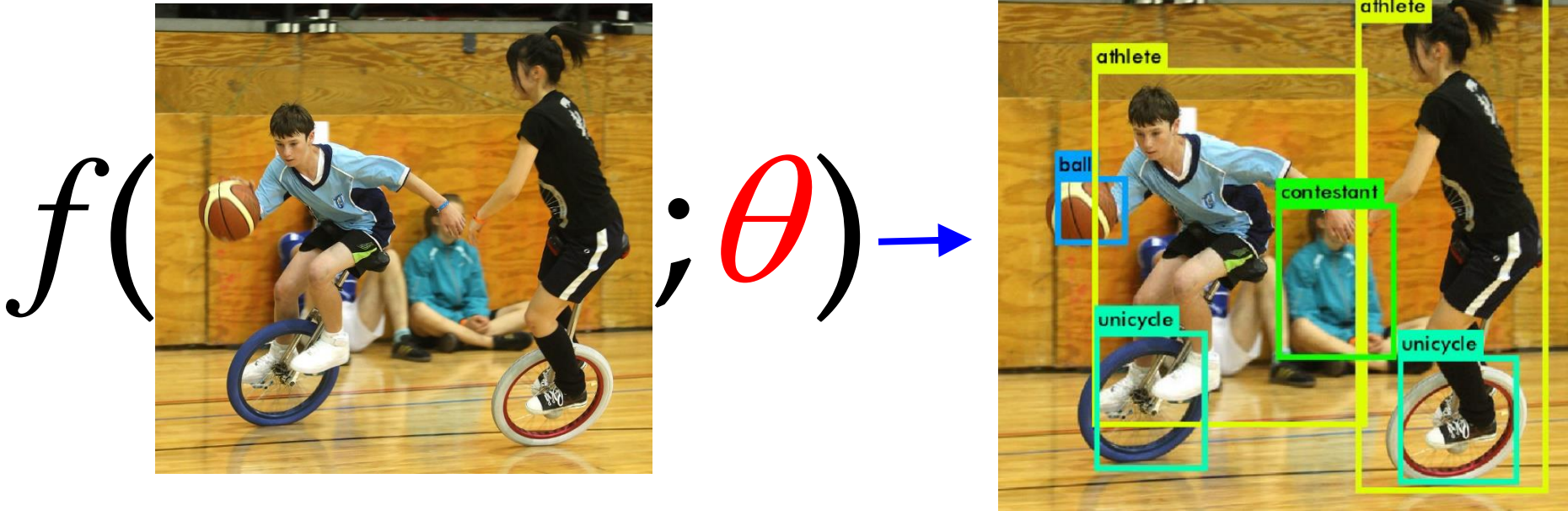
Training classes (757)



130,000 images
d'entraînement

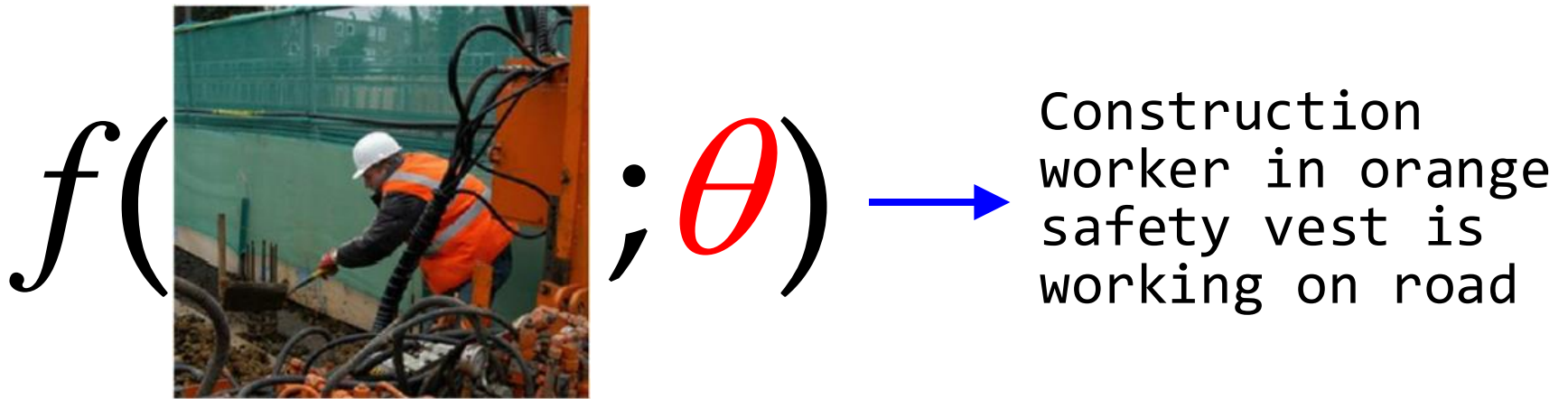


Détection d'objets



Redmon and Farhadi, YOLO9000: Better, Faster, Stronger, CVPR 2017.

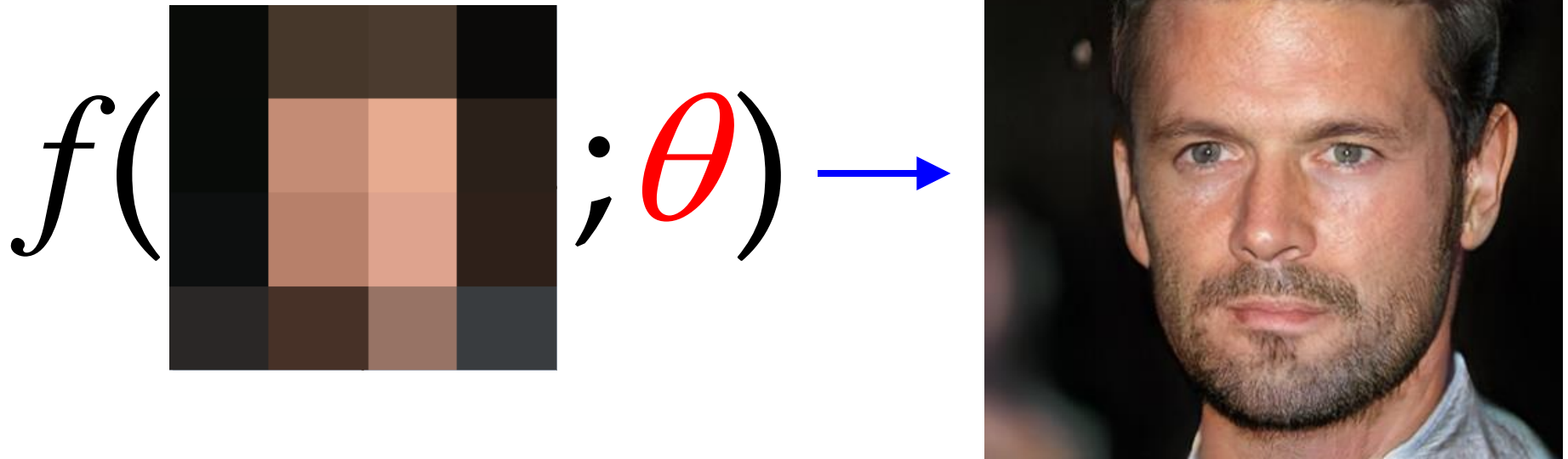
Description d'images



Qui est cet acteur?



Génération d'images





Karras et al., Progressive Growing of GANs for improved quality, stability and variation, submitted to ICLR 2018.

Reconnaissance voix

$f(\text{waveform}; \theta)$ → Ok Google,
where is my
car

Génération de voix : *Tacotron 2*

$f(\text{She earned a doctorate in sociology at Columbia University}; \theta)$  

Note : exemple d'entraînement : 

« George Washington was the first President of the United States »

A: 

B: 

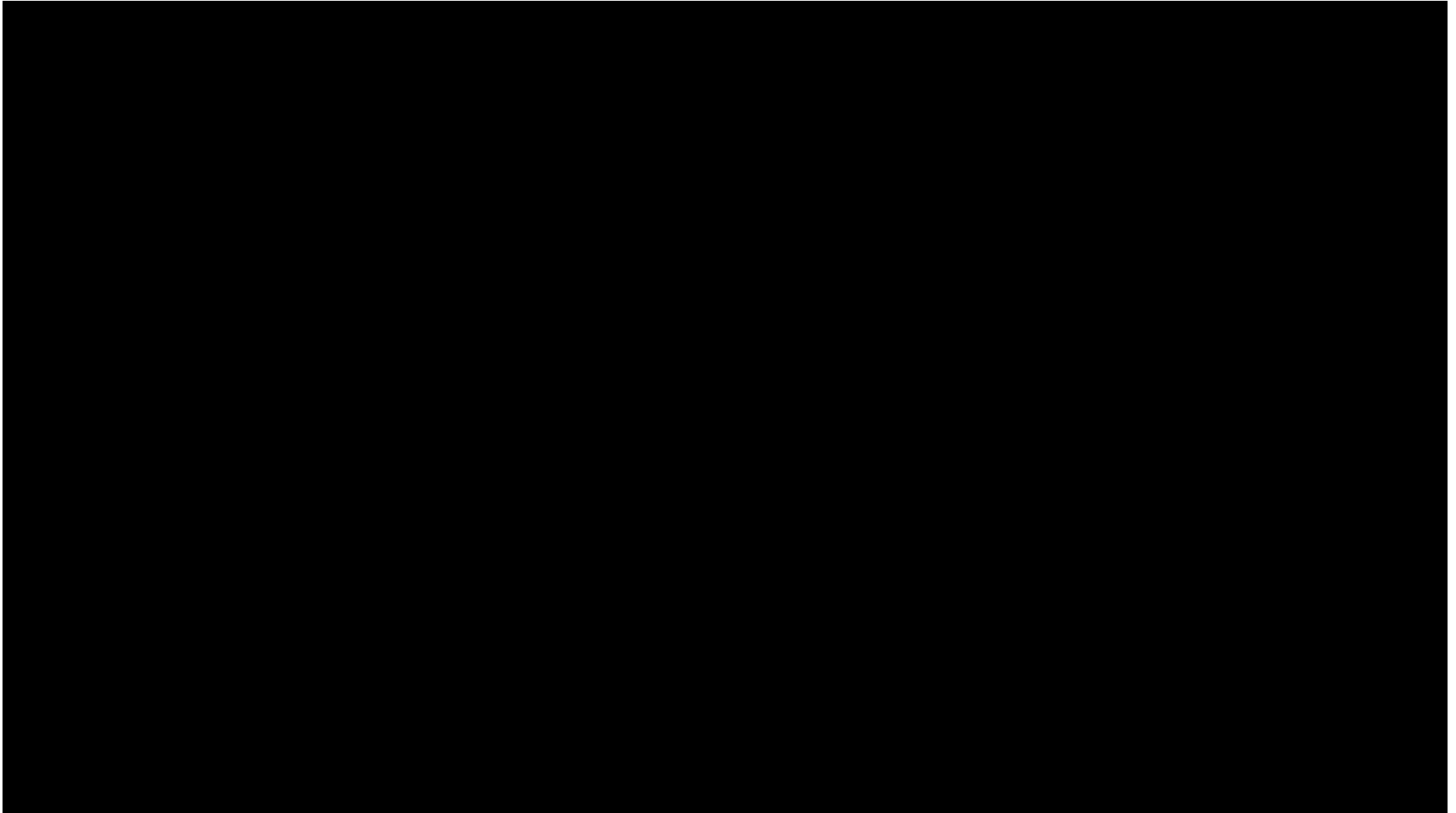
Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions

[Jonathan Shen](#), arXiv:1712.05884, dec 2017.

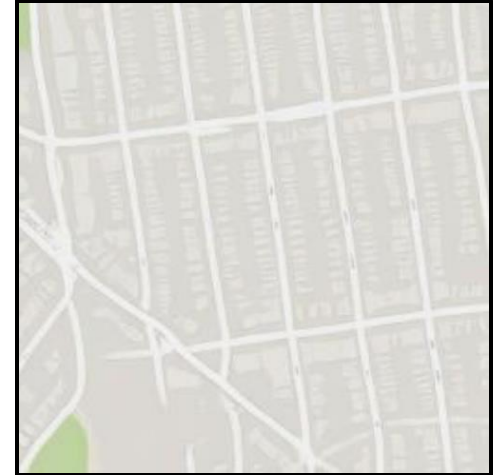
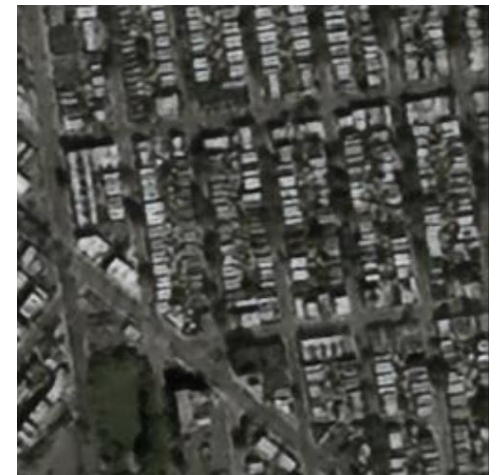
Traduction automatique

$f(\text{I think, therefore I am.}; \theta) \rightarrow \text{Je pense donc je suis.}$

Apprentissage visuomoteur



Transfert de style

 $G($  $; \theta)$  $F($ $; \theta)$ 

Zhu et al., Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, CVPR 2017.

Transfert de style

Monet \leftrightarrow Photos



Monet \rightarrow photo

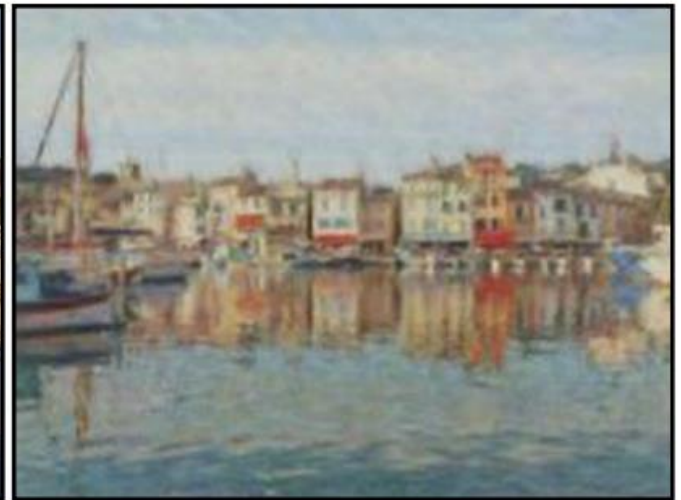


photo \rightarrow Monet

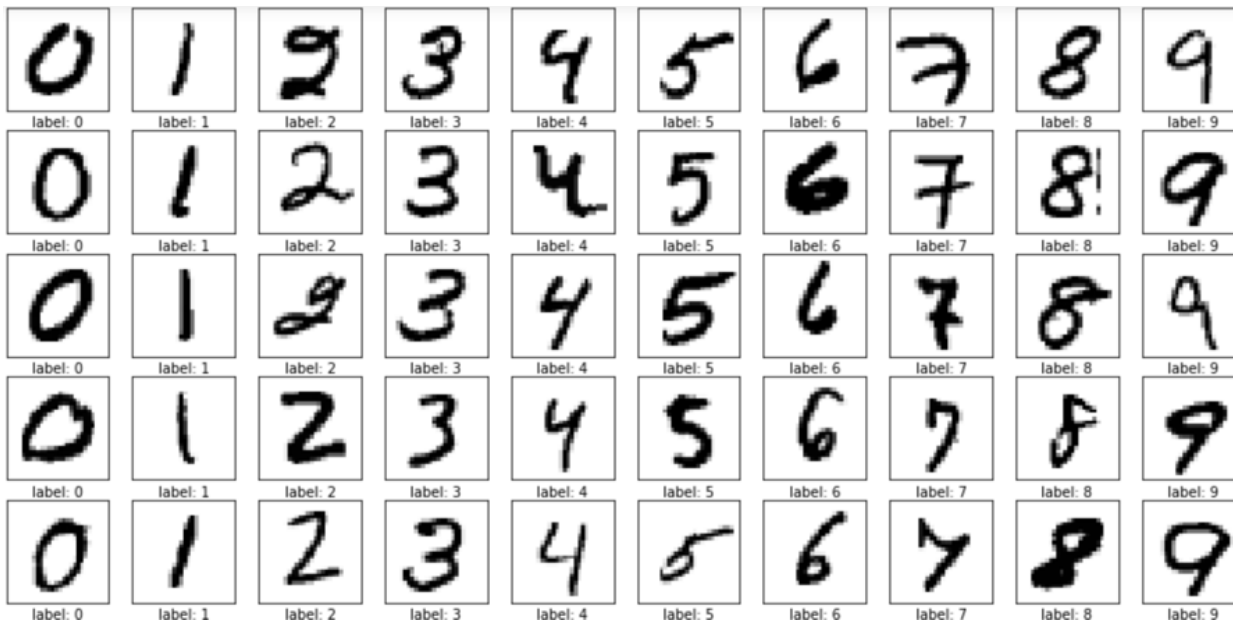
Jeu



Réseaux classiques
peu profonds :
classification d'images

MNIST

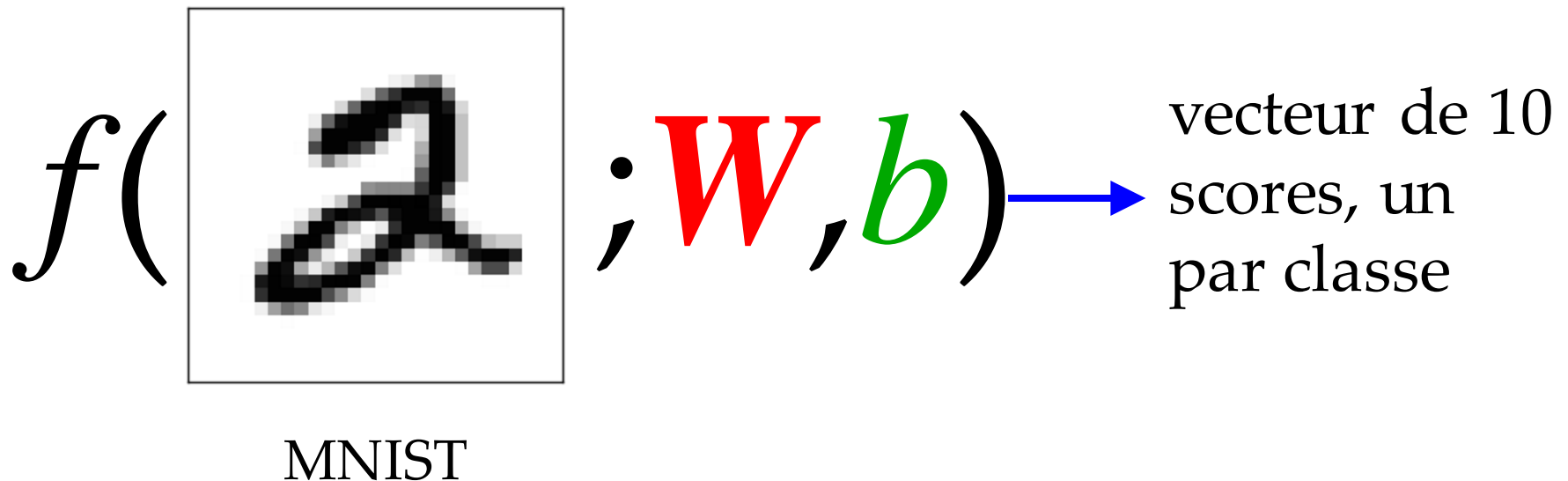
- 10 chiffres
- 70,000 images de 28×28 pixels



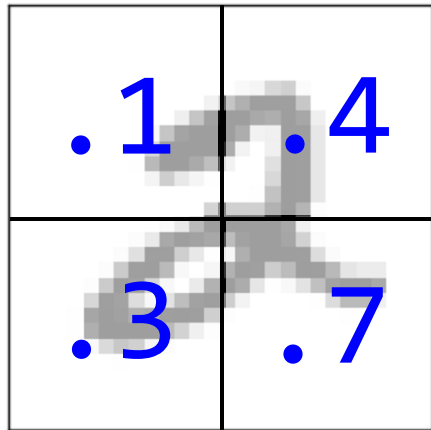
Source : <http://blog.welcomege.com/mnist-database/>

Classificateur linéaire sur MNIST

- Le plus simple possible : **linéaire**
- Paramètres **W** , taille $c \times n$ + biais **b** , taille $c \times 1$
 - n : nombre de pixels
 - c : nombre de classes



Exemple simplifié



« Flatten »

W

.7	-.3	.2	.3
-.5	1.7	1.5	.4
.7	1.1	-.4	-.1

.1
.4
.3
.7

+

b

1
1.2
-.4

=

Score

1.22
2.56
-.08

Chiffre 1

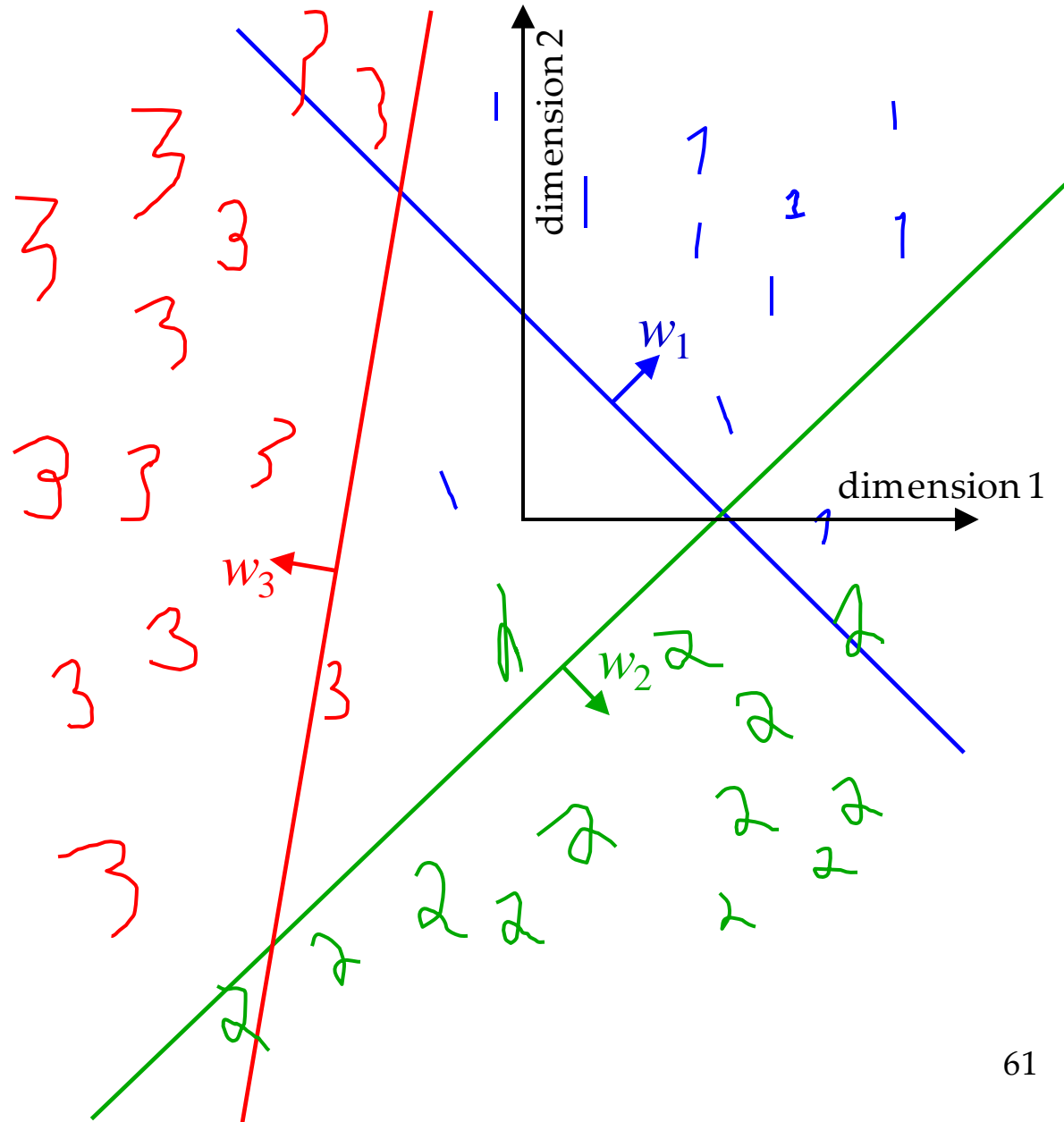
Chiffre 2

Chiffre 3

Interprétation classificateur linéaire

$$f = \mathbf{W}x + b$$

w_1	.7	-.3	.2	.3
w_2	-.5	1.7	1.5	.4
w_3	.7	1.1	-.4	-.1



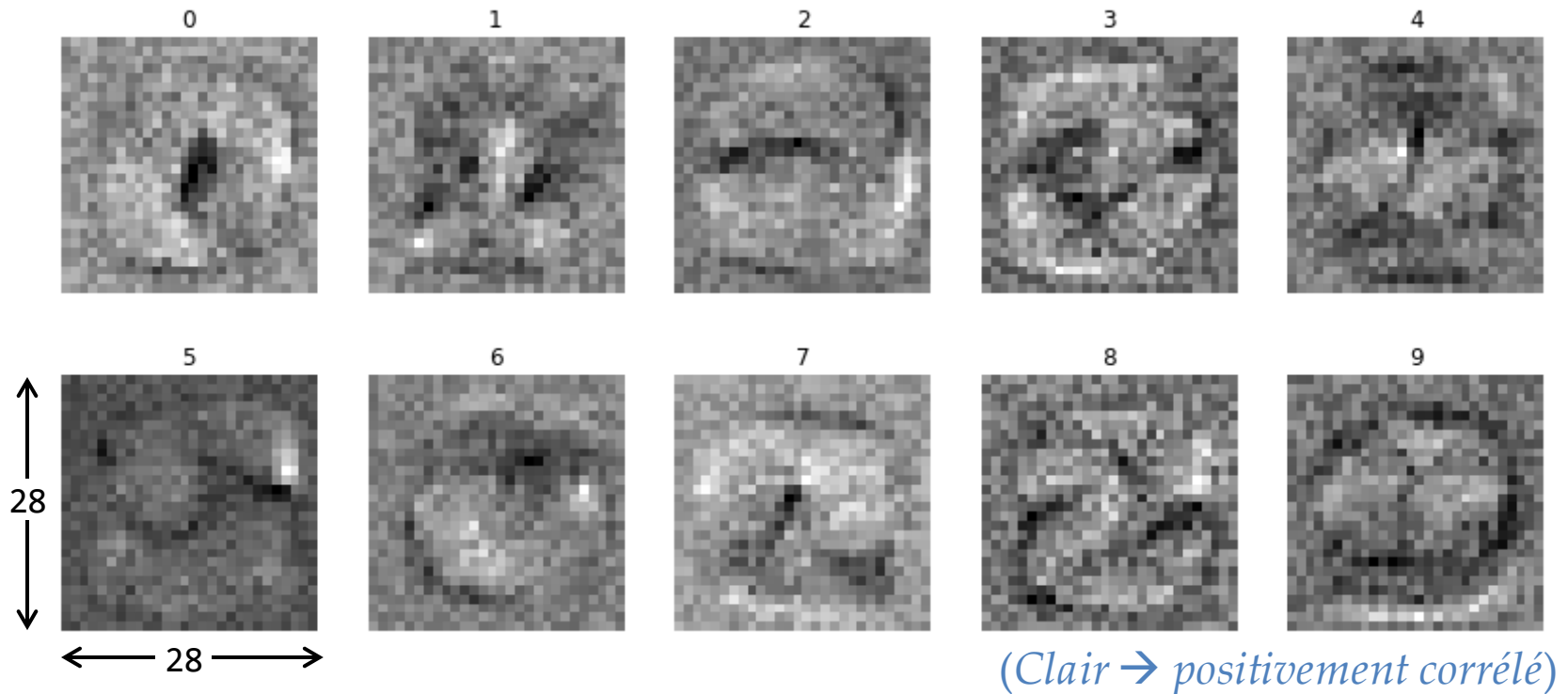
Réseau 1 couche linéaire

- Image 28x28 → 784x1
- Matrice **W** 10x784
- Biais **b** 10x1
- Initialisé au hasard
- Entraînement SGD sur
perte *multiclass hinge loss*
- Train set : 60,000 exemples
- Test set : 10,000 exemples
- Résultat ~92% précision sur test

```
class SimpleLinear(nn.Module):  
    def __init__(self):  
        super(SimpleLinear, self).__init__()  
        self.fc1 = nn.Linear(784, 10)  
  
    def forward(self, x):  
        x = x.view(-1, 784) # Flatten  
        x = self.fc1(x)  
        return x
```

$$L = \frac{1}{m} \sum_{i \neq \text{cible}} \max(0, 1 - \overbrace{\text{sortie}(\text{cible})}^{+ \text{grand}} + \underbrace{\text{sortie}(i)}_{+ \text{petits}})$$

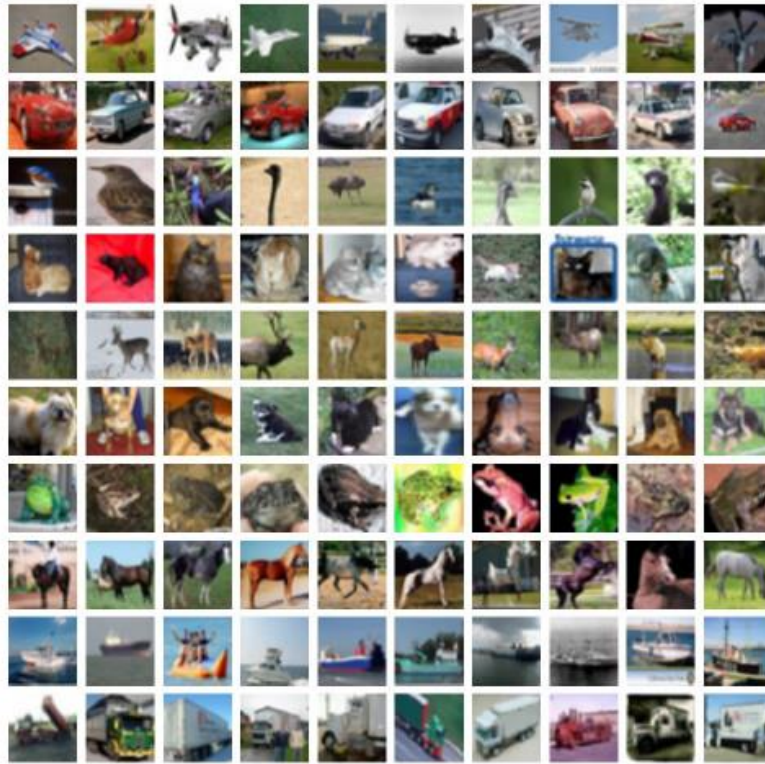
Poids W appris : patrons 28×28



- On voit corrélation spatiale sur pixels voisins
- Réseau doit la découvrir : doit avoir + d'exemples d'entraînement
- Si applique permutation identique sur les pixels, même précision

Sur images couleur CIFAR-10

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck



$$32 \times 32 \times 3 = 3072$$

Image RGB

				160	44	84	137
				81	94	119	36
90	44	17	63				
101	78	11	201				
53	41	98	42				
99	32	39	143				

Exemple de W trouvé



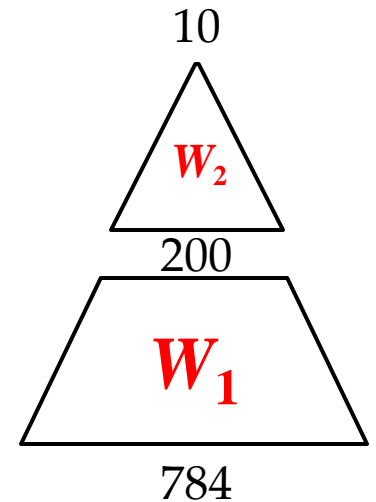
(Note : biais dans le dataset : l'image est centrée sur l'objet)

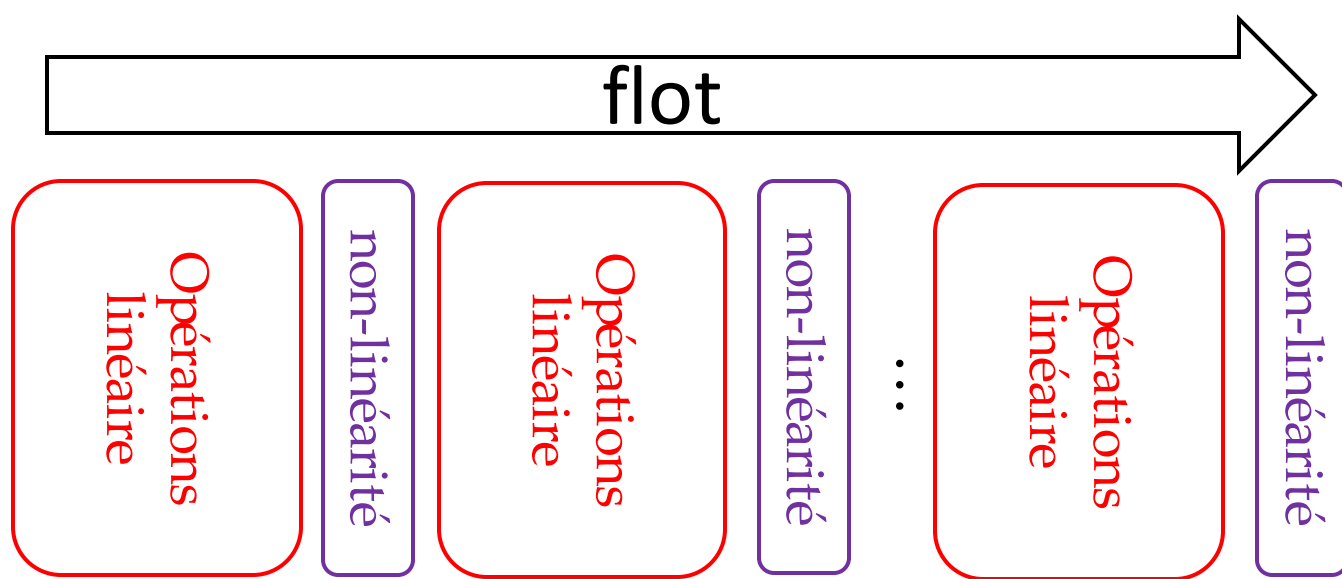
Réseau 2 couches linéaires

- Matrice W_1 200x784, bias b_1 200x1
- Matrice W_2 10x200, biais b_2 10x1
- # paramètres ~160,000
 - précédent ~8,000 paramètres
- Résultats? Encore ~92%!
- Pas plus expressif que linéaire 1 couche, car se simplifie :

$$W_2(W_1x + b_1) + b_2 = Wx + b$$

- Il faut ajouter non-linéarités pour augmenter la puissance d'expression





Ajout de
non-
linéarité

Pré-activation : $z = b + \sum_i w_i x_i = b + w^T x$

Activation : $h = g(z) \leftarrow \text{non-linéaire}$

