

Apprentissage profond

Introduction à l'apprentissage automatique – GIF-4101 / GIF-7005

Professeur : Christian Gagné

Semaine 8



8.1 Motivations de l'apprentissage profond

Historique des réseaux de neurones

- 1957 : proposition du perceptron par Frank Rosenblatt
- 1967 : démonstration par Marvin Minsky que le perceptron est incapable de traiter des données non linéairement séparables, désintérêt pour les approches neuronales
- 1986 : Rumelhart, Hinton et Williams démontrent l'utilisation de la rétropropagation des gradients pour l'entraînement du perceptron multicouche
- 1995-2005 : développement des SVM, perte d'intérêt pour les réseaux de neurones
- 2006 : premières architectures profondes de réseaux de neurones
- 2012 : résultats en reconnaissance d'objets (Toronto, ImageNet) et de la parole (Microsoft) démontre le potentiel de technologie disruptive de l'apprentissage profond
- 2014 : explosion d'investissements privés en apprentissage automatique, en particulier en apprentissage profond

Émergence des réseaux profonds

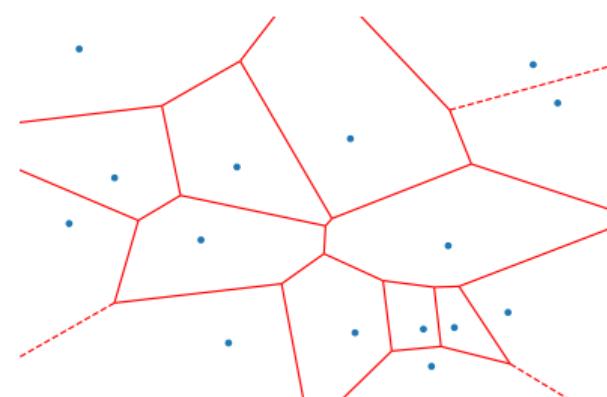
- Conditions ayant permis l'émergence des réseaux profonds :
 1. Disponibilités de très grands jeux de données (*big data*)
 2. Disponibilité d'une capacité de calcul faramineuse (GPU)
 3. Nouveaux modèles d'apprentissage très flexibles, avec des *a priori* permettant de bien gérer la malédiction de la dimensionnalité

Composition de modèles

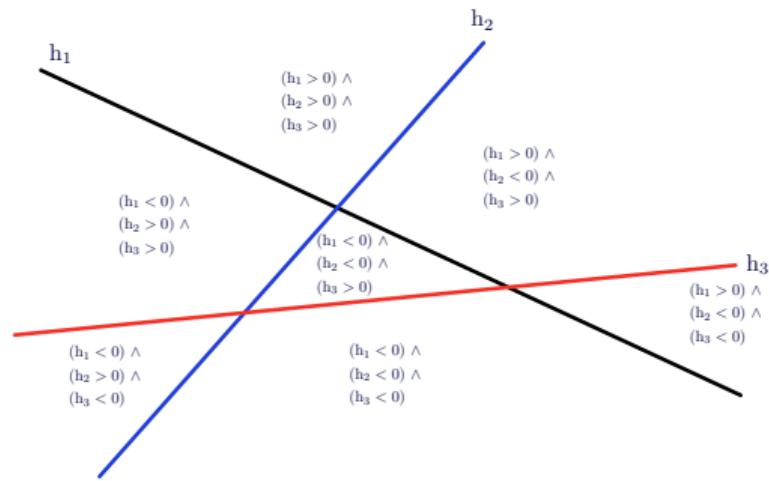
- « Compositionnalité » de modèle est nécessaire en apprentissage automatique
 - Comme en langue naturelle, on compose des éléments simples permettant d'exprimer des notions complexes
- Exploiter la compositionnalité permet un gain exponentiel en puissance de représentation
 - Représentations distribuées, apprentissage de caractéristiques
 - Architectures profondes : plusieurs niveaux d'apprentissage de représentations
- Composition de modèles est utile pour décrire notre monde efficacement

Représentation locale vs distribuée

- Ensemble de discriminants distribués (non mutuellement exclusifs) est exponentiellement plus efficace sur le plan statistique que des représentations locales (k -plus proches voisins, clustering)



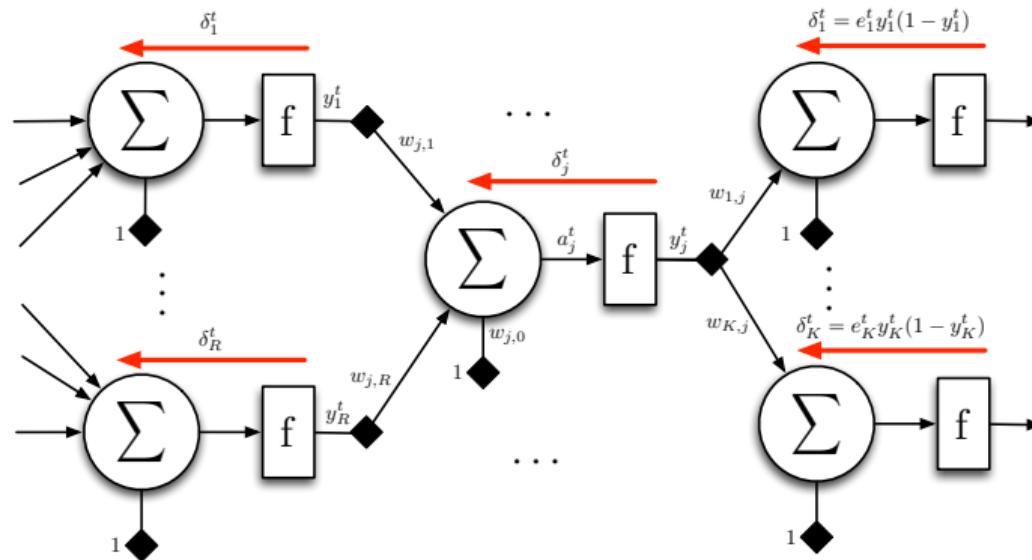
Représentation locale



Représentation distribuée

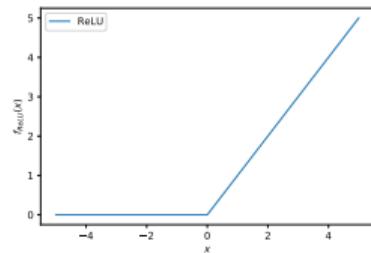
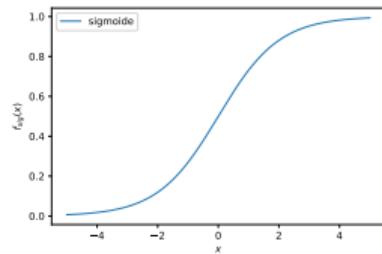
Problème de la dilution du gradient

- L'entraînement de perceptron multicouche de plus de deux couches cachées avec rétropropagation ne fonctionne pas bien
 - Neurones saturés, avec gradient très faible
 - Dilution du gradient (*vanishing gradient*) de couche en couche



Fonctions de transfert

- Fonction sigmoïde
 - Interprétation probabiliste
 - Approximation d'une fonction *step* (binaire)
 - Problème de saturation sur le gradient
- Fonctions de transfert doivent inclure des non-linéarités
- Fonction ReLU (*Rectified Linear Unit*),
 $f_{\text{ReLU}}(a) = \max(0, a)$
 - Modèle simple de fonction de transfert avec non-linéarité
 - Composition de ReLU permet de l'approximation linéaire par morceaux
 - Motivation biologique de réseaux profonds avec ReLU (*leaky integrate-and-fire model*)
 - Apprentissage de réseaux profonds avec ReLU possible sans pré-entraînement non supervisé



Profondeur des réseaux

- Les réseaux profonds, lorsque bien entraînés, apprennent mieux que les réseaux obèses (*fat networks*)
 - Capacité des réseaux croît linéairement selon la largeur d'une couche, exponentiellement selon la profondeur du réseau

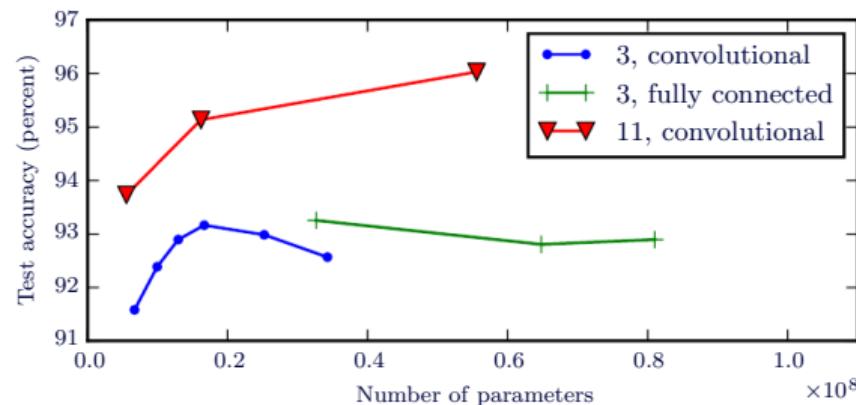


Figure 6.7 de I. Goodfellow, Y. Bengio et A. Courville, *Deep Learning*, MIT Press, 2016. Accédée en ligne le 19 octobre 2020 au <https://www.deeplearningbook.org/contents/mlp.html>.

- Réseau obèse fait du surapprentissage à 20M de poids, réseau profond fonctionne bien avec 60M de poids

8.2 Autoencodeurs et RBM

Pré-entraînement non supervisé

- Réseaux profonds avant 2011 : pré-entraînement non supervisé nécessaire
 - Initialisation aléatoire de réseaux profonds génère une grande variété de solutions sous-optimales (minima locaux)
 - Pré-entraînement non supervisé permet de démarrer la rétropropagation dans une « bonne configuration » (bassin d'attraction)

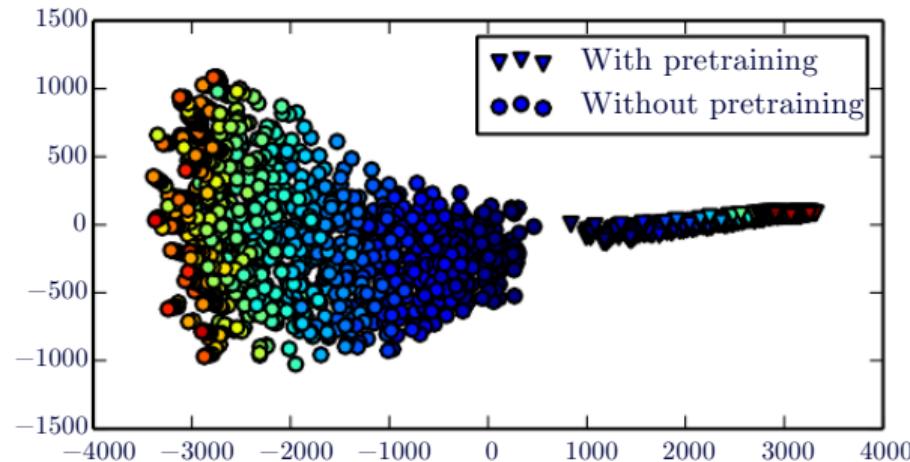
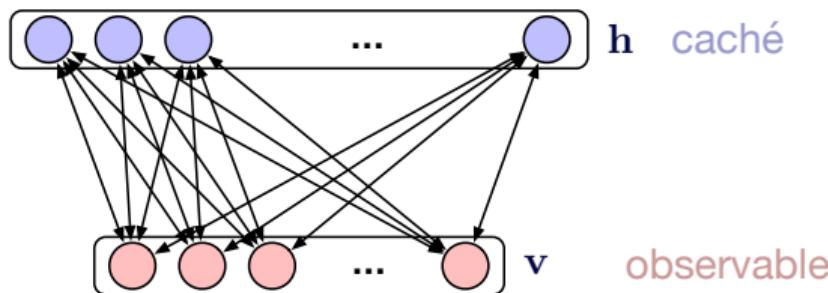


Figure 15.1 de I. Goodfellow, Y. Bengio et A. Courville, Deep Learning, MIT Press, 2016. Accédée en ligne le 19 octobre 2020 au <https://www.deeplearningbook.org/contents/representation.html>.

Machine de Boltzmann restreinte

- Machine de Boltzmann restreinte (RBM) : modèle génératif de réseau de neurones
 - Peut apprendre des distributions sur les données d'entrées
 - Couche de neurones visibles (v) et de neurones cachées (h)



- h et v sont binaires, modèle permet de calculer $P(v,h)$, $P(v)$, $P(v|h)$, $P(h|v)$
- Utilisé pour apprendre *deep belief network*, avec apprentissage non supervisé par couche de RBM, suivi d'un raffinement par rétropropagation des erreurs (supervisé)

Machine de Boltzmann restreinte

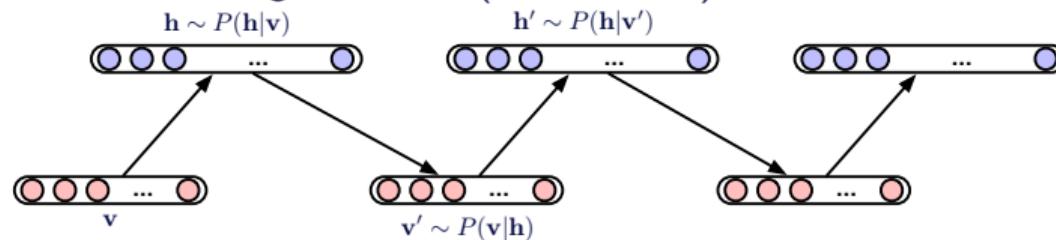
- Fonction d'énergie des RBM

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{a}^\top \mathbf{v} - \mathbf{b}^\top \mathbf{h} - \mathbf{v}^\top \mathbf{W} \mathbf{h}$$

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp[-E(\mathbf{v}, \mathbf{h})]$$

- Z est une fonction de partition, permet de normaliser les valeurs pour que les probabilités somment à 1

- Calcul de Z , et donc $P(\mathbf{v}, \mathbf{h})$, est intractable ($Z = \sum_{\forall \mathbf{v}, \mathbf{h}} \exp[-E(\mathbf{v}, \mathbf{h})]$)
- Solution : échantillonnage de Gibbs (Monte Carlo)

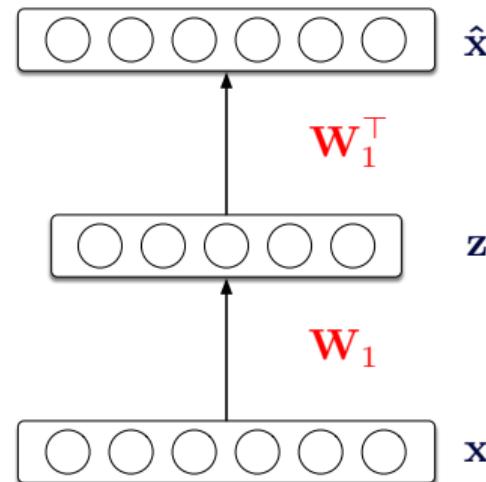


- Computationnellement très lourd

- RBM très peu utilisées de nos jours pour réseaux profonds

Autoencodeurs

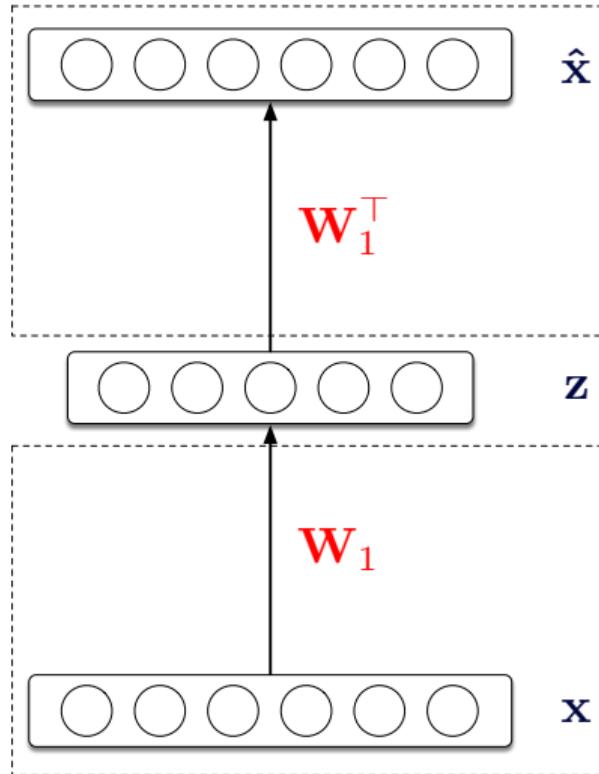
- Autoencodeur : modèle permettant de faire une compression de l'entrée (encodeur) et une décompression de celle-ci (décodeur)
 - Objectif : compresser tout en gardant l'erreur $\|\mathbf{x} - \hat{\mathbf{x}}\|^2$ faible
 - Poids du décodeur liés aux poids de l'encodeur (habituellement, transposé)



Entraînement d'autoencodeurs

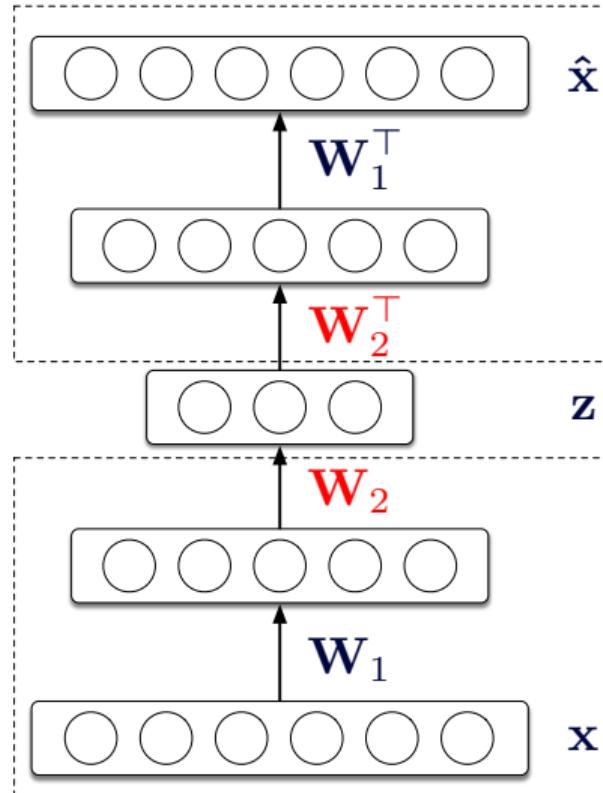
- Autoencodeur entraîné de façon non supervisée, pour apprendre représentation
 - Encodeur utilisé pour extraire une représentation compacte
- Entraînement vorace, une couche à la fois
 - Entraînement de la couche la plus externe
 - Ajout d'une nouvelle couche, qui est entraînée individuellement, couche externe étant fixée, et ainsi de suite
- Fonction de transfert non linéaire entre les couches
 - Nécessaire, sinon plusieurs couches linéaires pourraient se simplifier en une seule couche
 - Apprentissage des poids par descente du gradient
- Couche de sortie ajoutée à l'encodeur, avec entraînement supervisé
 - Entraînement complet de la couche de sortie par rétropropagation
 - Ajustement des poids de l'encodeur par rétropropagation (*fine-tuning*)

Exemple d'entraînement d'un autoencodeur



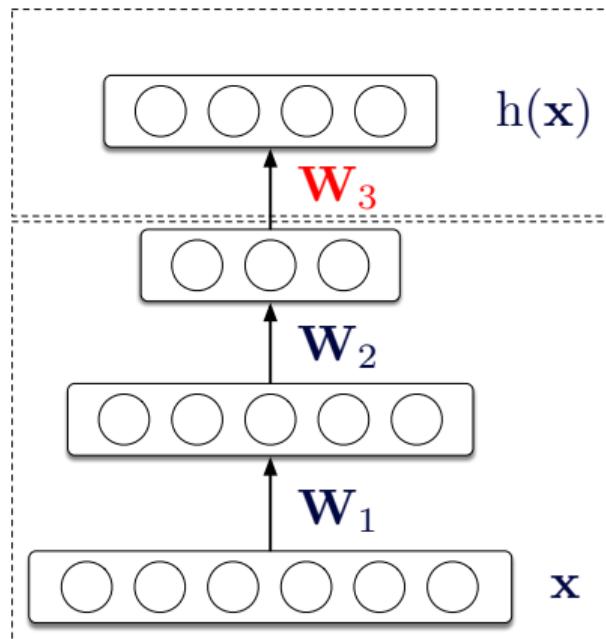
- Entraînement non supervisé du poids W_1 , poids W_1^T lié
- Minimize erreur $\|x - \hat{x}\|^2$
- Représentation intermédiaire dans valeurs centrales

Exemple d'entraînement d'un autoencodeur



- Ajout de deux nouvelles couches (une dans encodeur et une dans décodeur)
- Entraînement non supervisé du poids W_2 , poids W_1 fixés
- Minimise toujours erreur $\|x - \hat{x}\|^2$
- Nouvelle représentation intermédiaire
- Peut être répété ainsi sur plusieurs couches

Exemple d'entraînement d'un autoencodeur



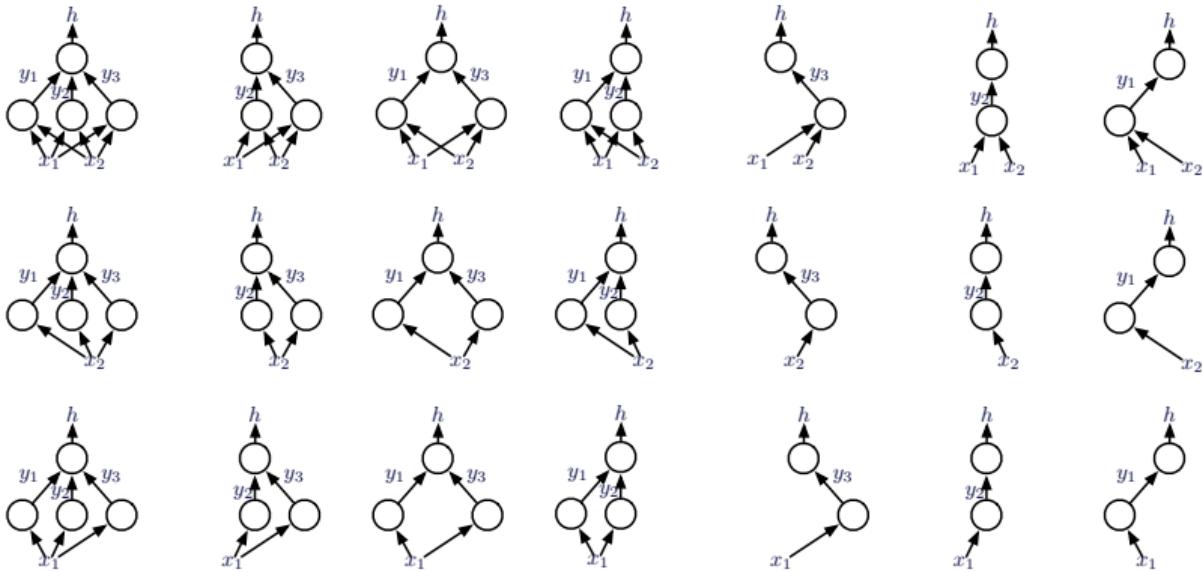
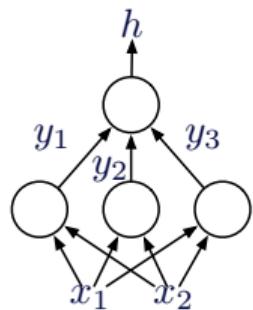
- Retrait de la partie décodeur du réseau
- Ajout d'une couche de sortie, avec autant de sorties que de classe
- Entraînement **supervisé** de W_3 par rétropropagation
- Poids W_1 et W_2 souvent également ajustés finement par rétropropagation (*fine-tuning*)

8.3 Dropout et batch normalization

Dropout

- Dropout : méthode d'entraînement par désactivation aléatoire des neurones
 - Typiquement, moitié des neurones des couches cachées (80 % des entrées) sont activés à la présentation de chaque donnée durant l'entraînement
 - Masques aléatoires pour sélectionner neurones actifs, un différent à chaque présentation
- Effectue une régularisation du réseau
 - Force l'apprentissage d'une représentation distribuée dans l'ensemble du réseau
 - Rend difficile l'émergence de « neurones grand-mère »
 - S'est avéré très efficace pour améliorer les performances des réseaux profonds
- Évaluation de nouvelles données en test par moyennage sur plusieurs masques de sélection
 - Analogie avec méthodes par ensemble (vu plus tard dans la session), en particulier bagging

Dropout



Batch normalization

- Modification d'un poids par rétropropagation basée sur gradient local
 - Poids des couches précédentes et suivantes eux aussi modifiés !
- *Batch normalization* : normaliser activation des neurones entre toutes les données d'un mini-lot (*mini-batch*)
 - Mini-lot : petit sous-ensemble d'instances de données de l'ensemble d'entraînement (typiquement quelques centaines)
- Activation des neurones \mathbf{H} normalisées selon

$$\mathbf{H}' = \frac{\mathbf{H} - \mu}{\sigma}, \quad \mu = \frac{1}{m} \sum_i \mathbf{H}_{i,:}, \quad \sigma = \sqrt{\epsilon + \sum_i (\mathbf{H} - \mu)_i^2}$$

- \mathbf{H} : activation des neurones (ligne) d'une couche pour les données du mini-lot (colonne)
- ϵ : petite valeur (typiquement 10^{-8}) pour éviter division par zéro lorsque variance nulle

8.4 Traitement de texte et d'images

Traitement de texte

- Comment donner des documents (séquence de chaînes de caractères) à un réseau de neurones (vecteur de réels de taille fixe) ?
- Modèle *Bag-of-Words* (BoW)
 - Identifier dictionnaire de mots les plus fréquents / intéressants
 - Calculer la fréquence de chaque mot pour chaque document traité (vecteur d'entiers de taille fixe)

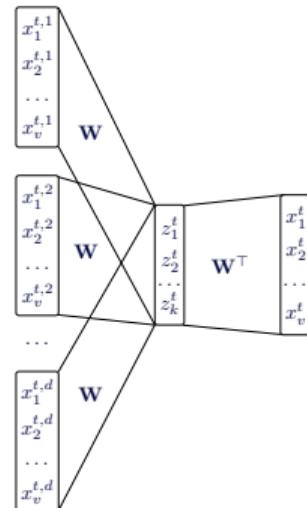
$$\mathbf{x}^t = [x_1^t, x_2^t, \dots, x_v^t]^\top$$

où x_i^t est le nombre d'occurrences (valeur entière) du i -ème mot (selon le dictionnaire) dans le document

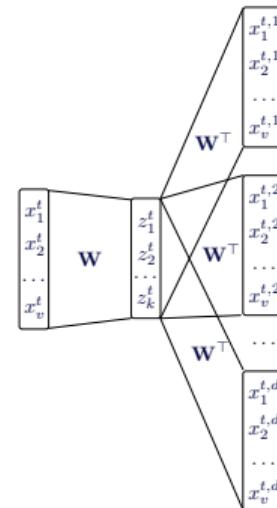
- Ne tient pas compte de l'ordre
 - Modèles avec N-gram mesure fréquence de groupes de mots adjacents
 - Skip-gram : mots connexes peuvent ne pas être adjacents

Représentations neuronales pour le traitement de texte

- Tenir compte de la séquence pour encoder le texte
 - *Continuous BoW* : prédire le mot selon ceux qui précédent et suivent (plus rapide)
 - *Continuous skip-gram* : prédire les mots qui précédent et suivent selon le mot d'intérêt (plus précis)



Continuous BoW

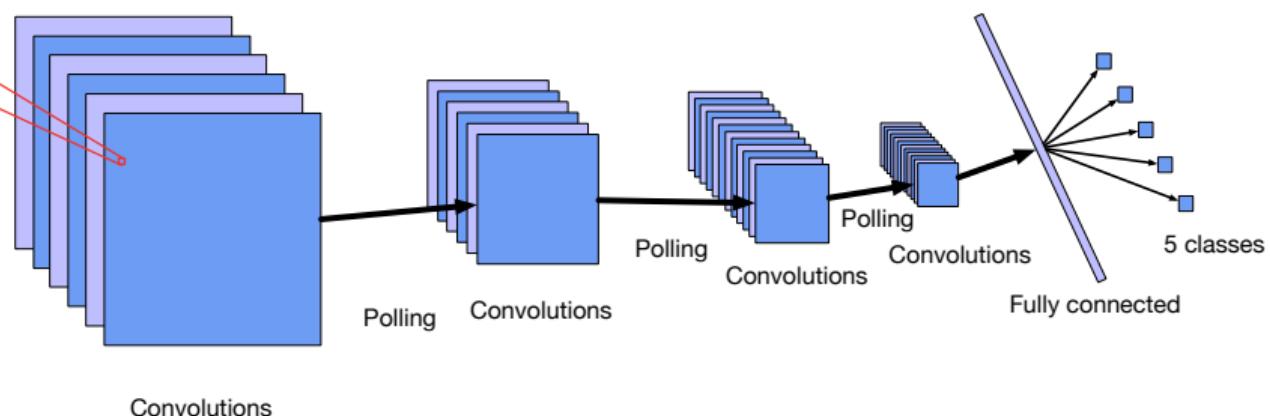
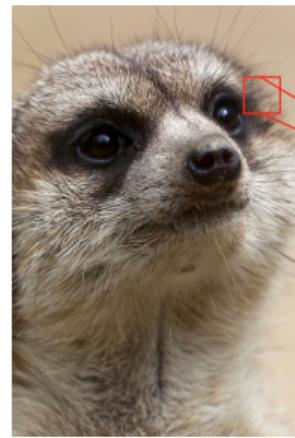


Continuous skip-gram

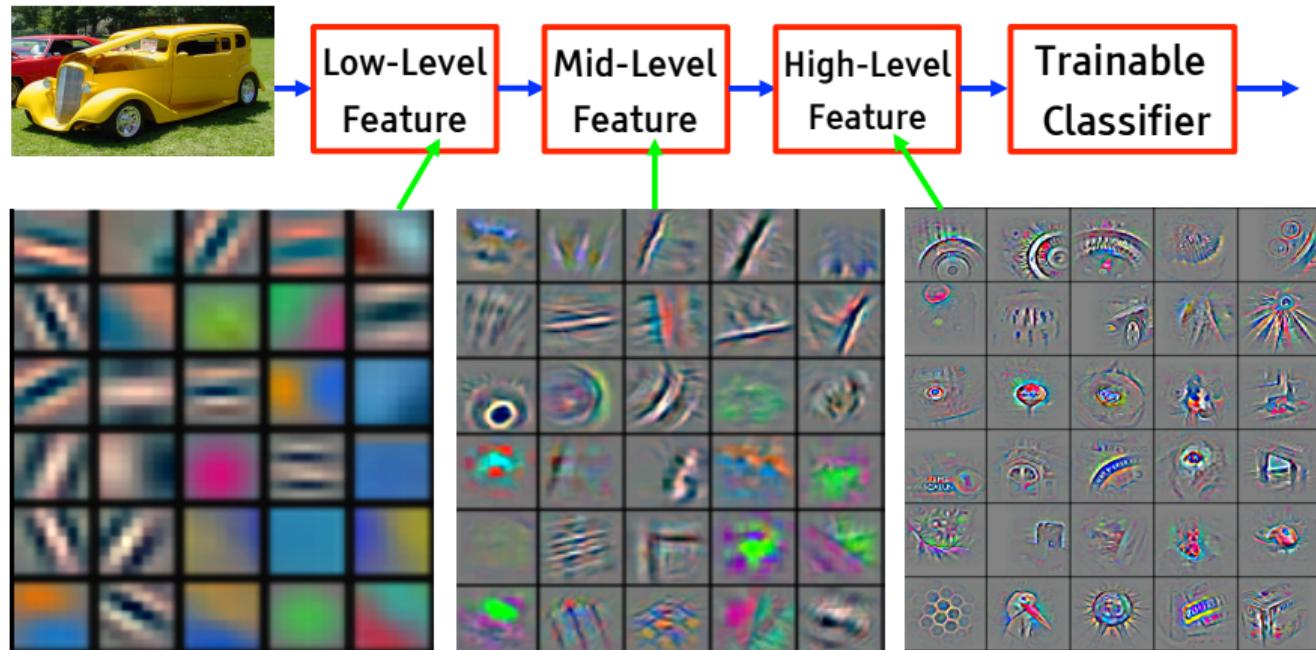
Réseau à convolution

- Réseau à convolution : traiter des signaux temporels ou spatiaux
 - Signal temporel : son et parole
 - Signal spatial : image
- Couche de convolution : filtres convolués sur données temporelles/spatiales
 - Données peuvent être valeurs d'entrée du réseau ou sorties de couches précédentes
 - Convolution sur chaque canal (plusieurs canaux possibles)
 - Apprentissage des filtres par rétropropagation
- Couche de *pooling* : sélection de valeurs (maximum d'une fenêtre)
 - Permet de réduire taille des valeurs, sinon explosion de la taille du modèle en vue !
- Neurones pleinement connectés en sortie pour prise de décision
- Présenté plus en détail la semaine prochaine

Réseau à convolution



Composition de filtres



Tiré de G. Hinton, Y. Bengio et Y. LeCun, Deep Learning NIPS'15 Tutorial, 2015. Accédé en ligne le 19 octobre 2020 au <https://media.nips.cc/Conferences/2015/tutorials/slides/DL-Tutorial-NIPS2015.pdf>.

Reconnaissance d'objets

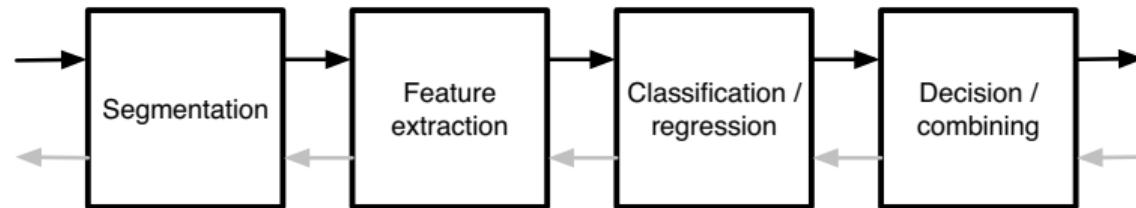
- *ImageNet Large Scale Visual Recognition Challenge* : reconnaître les objets d'images (1000 classes), en donnant la bonne classe dans un top 5

ILSVRC 2012		ILSVRC 2013		ILSVRC 2014	
Équipe	% erreur	Équipe	% erreur	Équipe	% erreur
SuperVision (Toronto)	15,3	Clarifai	11,7	GoogLeNet	6,6
ISI (Tokyo)	26,1	NUS	12,9	VGG (Oxford)	7,3
VGG (Oxford)	26,9	Zeiler-Fergus (NYU)	13,5	MSRA	8,0
XRCE / INRIA	27,0	A. Howard	13,5	A. Howard	8,1
UvA (Amsterdam)	29,6	OverFeat (NYU)	14,1	DeeperVision	9,5
INRIA / LEAR	33,4	UvA (Amsterdam)	14,2	NUS-BST	9,7
		Adobe	15,2	TTIC-ECP	10,2
		VGG (Oxford)	15,2	XYZ	11,2
		VGG (Oxford)	23,0	UvA	12,1

8.5 Apprentissage de représentations

Apprentissage de représentations

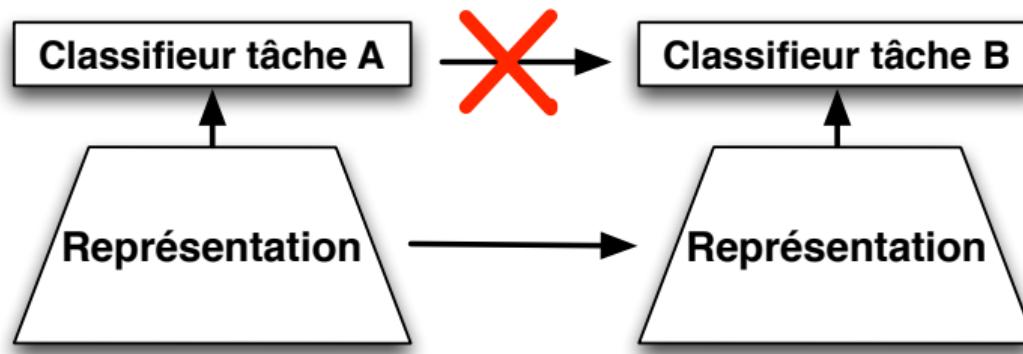
- Pipeline classique de la reconnaissance des formes



- Dans le passé, chaque module conçu indépendamment
- Apprentissage profond permet l'apprentissage de représentations
 - Apprentissage de tous les modules simultanément
 - Possibilité de récupérer les représentations (segmentation, extraction de caractéristiques) et les utiliser avec d'autres modules de classement et de prise de décision

Transfert de représentations

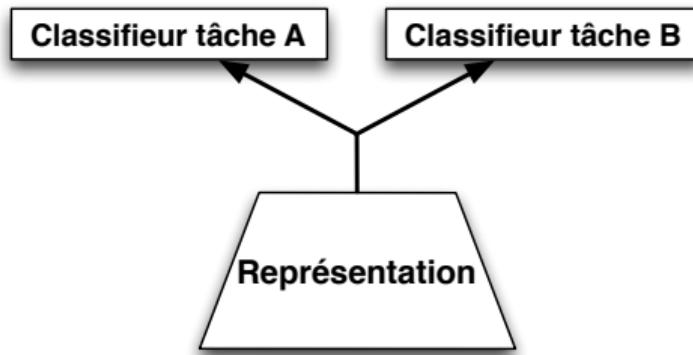
- Apprentissage d'un réseau profond sur tâche A
- Nouvelle tâche B, basée sur données similaires à tâche A
 - Récupérer représentation de tâche A
 - Entraîner nouveau classifieur pour tâche B



- Permet un transfert de représentation (*transfer learning*)

Apprentissage multitâche

- Apprentissage multitâche : apprendre *simultanément* une représentation pour des opérations distinctes
 - Réseau à deux têtes, une pour chaque tâche



- Rétropropagation provient d'une tête à la fois
- Mélange des données et des tâches durant l'apprentissage
- Performe bien à produire des représentations capturant des concepts généraux

8.6 Génération d'images

Génération d'exemples

- Idée : générer des données d'entrées à partir d'une sortie désirée
 - Générer donc un modèle de la donnée pouvant produire la sortie selon le réseau de neurones
- Approche : descendre le gradient sur la donnée d'entrée

$$\Delta \mathbf{x} = -\eta \frac{\partial E(\mathbf{x}|\theta)}{\partial \mathbf{x}}$$

- On va donc générer une nouvelle donnée à partir de la valeur initiale de \mathbf{x} et la sortie désirée \mathbf{r}
- Poids du réseau ne changent pas
- Utilisé dans diverses circonstances
 - Générateur d'images *Deep Dream* de Google

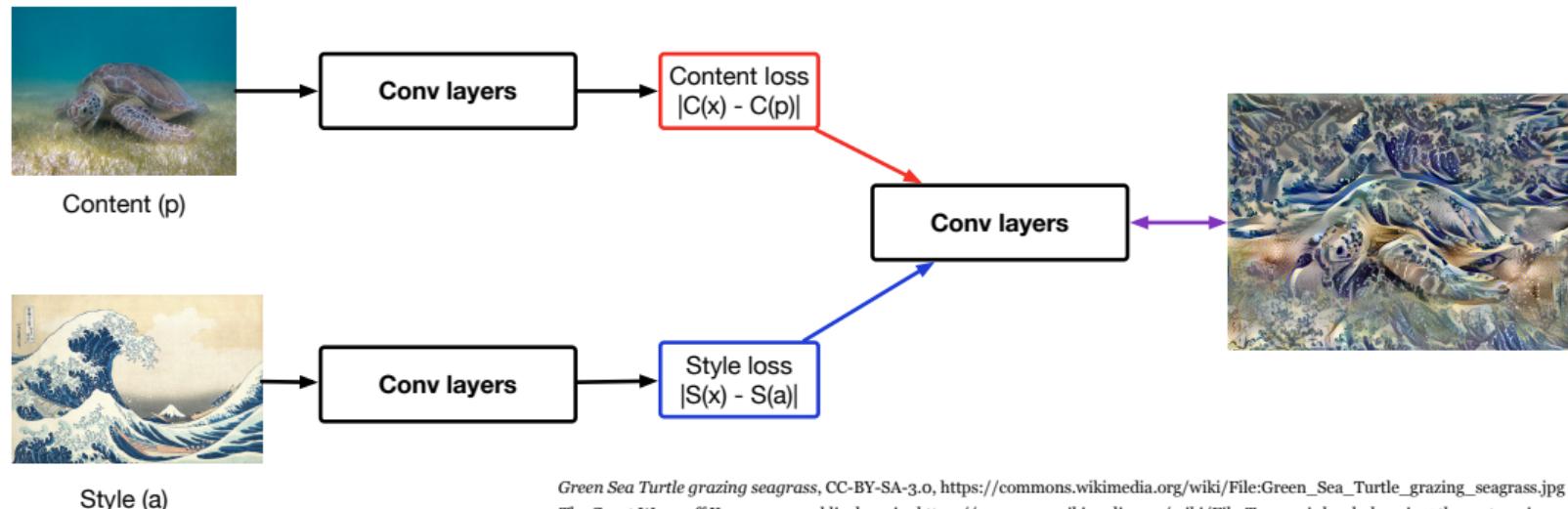
Deep dream



Par Google, CC-BY 4.0, <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

Transfert de style

- Idée : transférer le style d'une image dans une nouvelle image
 - Comparer le contenu dans les couches de convolution (ex. VGG19) et le style (matrice de Gram)



Green Sea Turtle grazing seagrass, CC-BY-SA-3.0, https://commons.wikimedia.org/wiki/File:Green_Sea_Turtle_grazing_seagrass.jpg
The Great Wave off Kanagawa, public domain, https://commons.wikimedia.org/wiki/File:Tsunami_by_hokusai_19th_century.jpg

8.7 Approches adversariales

Données adversariales

- Utiliser génération de données pour déterminer plus petite variation permettant de faire une erreur de classement

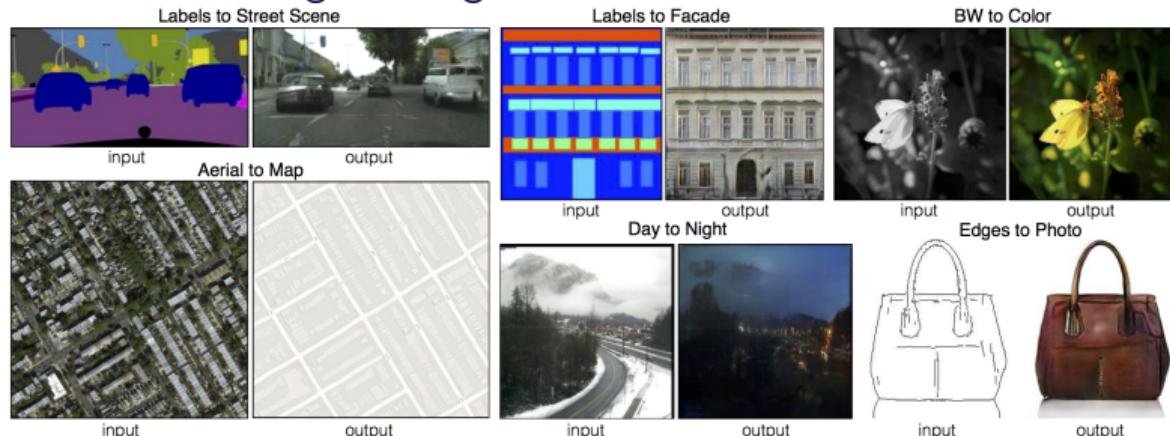


Meerkat $\epsilon = 0.005$ School bus
Conf.: 65.3% Conf.: 98.6%

- Causé par l'utilisation de représentation distribuée dans un espace à très haute dimensionnalité
- Illustre une difficulté actuelle avec réseaux profonds, robustesse aux données adversariales doit être améliorée

Generative Adversarial Networks (GAN)

- Modèle GAN : mettre en compétition deux réseaux de neurones
 - Réseau discriminatif : distinguer données véritables du problème des données générées
 - Réseau génératif : produire des données ayant l'air authentiques
 - Permet des traitements variés basés sur un apprentissage non supervisé
- Exemple : traduction image-à-image avec GANs conditionnels

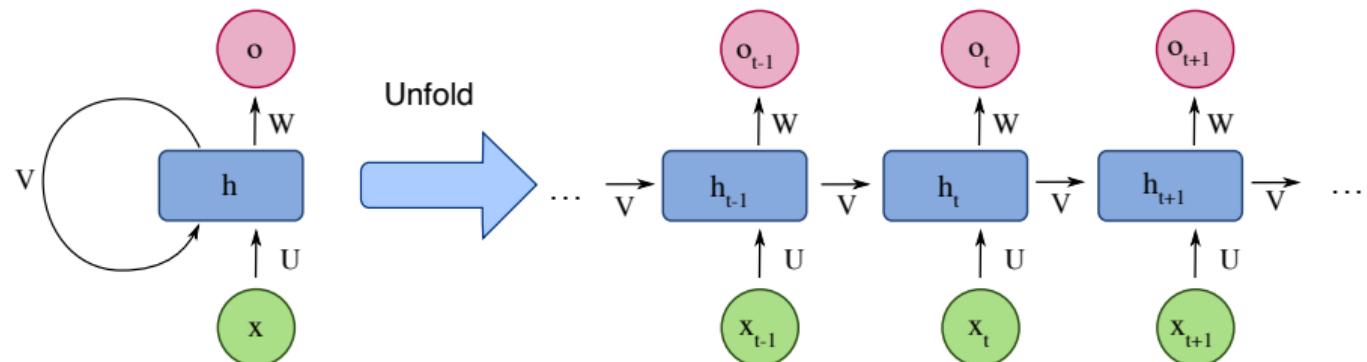


Tiré de *Isola, Zhu, Zhou et Efros, Image-to-Image Translation with Conditional Adversarial Networks, CVPR, 2017*. Accédé en ligne le 19 octobre 2020 au <https://arxiv.org/pdf/1611.07004v3.pdf>.

8.8 Traitement de séquences

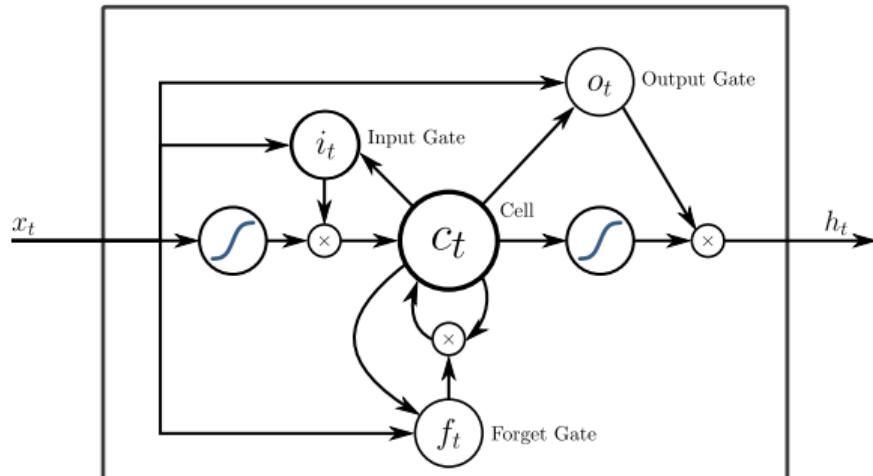
Réseau récurrent

- Réseaux usuels (*feedforward*) : données propagées dans le réseau, indépendant des données suivantes / précédentes
 - Traitement de données séquentielles important dans nombreux contextes
- Réseaux récurrents : connexions avec valeurs précédentes
 - Traitement avec algorithmes habituels en déroulant le réseau



Par fde洛che, CC-SA 4.0, https://commons.wikimedia.org/wiki/File:Recurrent_neural_network_unfold.svg

Long Short-Term Memory (LSTM)



Par Graves, Mohamed et Hinton, CC-SA 4.0, https://en.wikipedia.org/wiki/File:Peephole_Long_Short-Term_Memory.svg

- Modèle LSTM : ajouter de la mémoire au réseau
- Cellule de mémoire (état), avec quatre neurones
 - Entrée
 - Activation de l'entrée
 - Activation de l'oubli
 - Activation de la sortie

Apprentissage par renforcement profond

- Apprentissage par renforcement : déterminer les bonnes actions à effectuer selon les conditions actuelles
 - Guidé par récompense ponctuelle, sans indication précise sur actions décisives
 - Forme plus élaborée (et plus complexe) d'intelligence que tâches de classement et régression
- Réseaux profonds s'avèrent très prometteurs pour l'apprentissage par renforcement
 - Simulations massives permettent d'apprendre à effectuer certaines actions précises
- Jeux vidéos (Atari 2600) : Deep Q-learning Network (Deepmind)
 - Entrée est la capture de l'écran, récompense est le pointage obtenu
 - 49 jeux différents, performances « surhumaines »
- Jeu de Go : AlphaGo (encore Google Deepmind)
 - Go : jeu traditionnel asiatique, plus complexe que les échecs :
[http://www.theverge.com/2016/3/9/11185030/
google-deepmind-alphago-go-artificial-intelligence-impact](http://www.theverge.com/2016/3/9/11185030/google-deepmind-alphago-go-artificial-intelligence-impact)

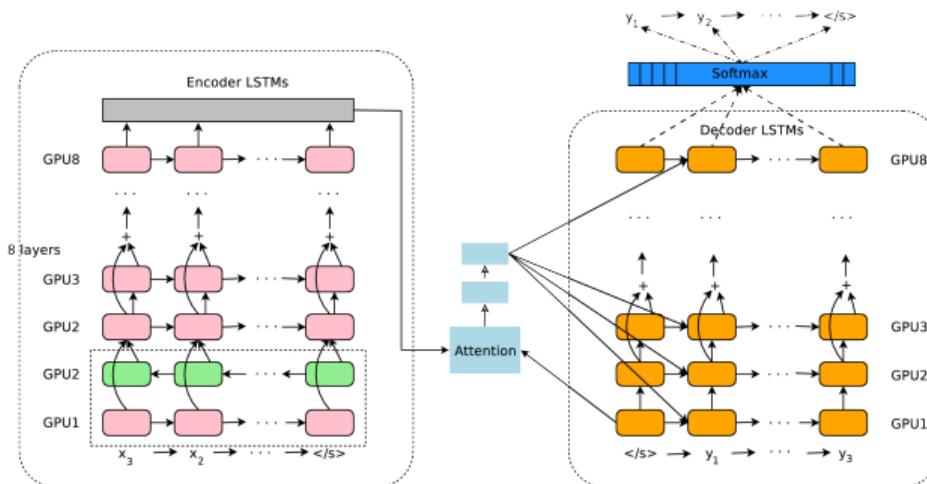
8.9 Applications de l'apprentissage profond

Conduite automatisée

- Véhicules autonomes : chamboulement prochain dans la façon de se déplacer
 - Premiers essais avec capteurs très coûteux (ex. LIDAR longue portée)
 - Développement avec technologies plus abordables (caméra vidéo, RADAR, Sonar)
- Grand potentiel de l'apprentissage profond pour conduite autonome
 - Détection d'objets et de piéton
 - Lecture de la signalisation (panneaux et lumières)
 - Interprétation de la conduite d'autres véhicules
 - Contrôle de la conduite
- Prédictabilité de l'apprentissage automatique, en particulier apprentissage profond, reste un problème
- Approche de Tesla pour la conduite autonome
 - Équiper tous les véhicules fabriqués de capteurs sophistiqués et d'un lien réseau
 - Collecter information sur la conduite par tous ces véhicules
 - Automatiser graduellement la conduite par un apprentissage sur ces données à mesure que la qualité des modèles appris augmente

Traduction automatisée

- *Google's Neural Machine Translation System* : nouvelle mouture de Google Translate basée sur des réseaux profonds
 - Amélioration de 60 % des performances relativement à la version précédente
 - Déployé sur les systèmes de Google simultanément à sa publication (automne 2016)



Tiré de Wu et al., *Google's Neural Machine Translation System : Bridging the Gap between Human and Machine Translation*, arXiv:1609.08144, 2016. Accédé en ligne le 19 octobre 2020 au <https://arxiv.org/pdf/1609.08144.pdf>.

Exemples de traduction avec GNMT

Source	When asked about this, an official of the American administration replied : "The United States is not conducting electronic surveillance aimed at offices of the World Bank and IMF in Washington."
PBMT (3.0)	Interrogé à ce sujet, un responsable de l'administration américaine a répondu : "Les États-Unis n'est pas effectuer une surveillance électronique destiné aux bureaux de la Banque mondiale et du FMI à Washington".
GNMT (6.0)	Interrogé à ce sujet, un fonctionnaire de l'administration américaine a répondu : "Les États-Unis n'effectuent pas de surveillance électronique à l'intention des bureaux de la Banque mondiale et du FMI à Washington".
Humain (6.0)	Interrogé sur le sujet, un responsable de l'administration américaine a répondu : "les États-Unis ne mènent pas de surveillance électronique visant les sièges de la Banque mondiale et du FMI à Washington".
Source	She was spotted three days later by a dog walker trapped in the quarry
PBMT (6.0)	Elle a été repéré trois jours plus tard par un promeneur de chien piégé dans la carrière.
GNMT (2.0)	Elle a été repérée trois jours plus tard par un traîneau à chiens piégé dans la carrière.
Humain (5.0)	Elle a été repérée trois jours plus tard par une personne qui promenait son chien coincée dans la carrière.

Tiré de Wu et al., *Google's Neural Machine Translation System : Bridging the Gap between Human and Machine Translation*, arXiv:1609.08144, 2016. Accédé en ligne le 19 octobre 2020 au <https://arxiv.org/pdf/1609.08144.pdf>.

8.10 Implémentations logicielles

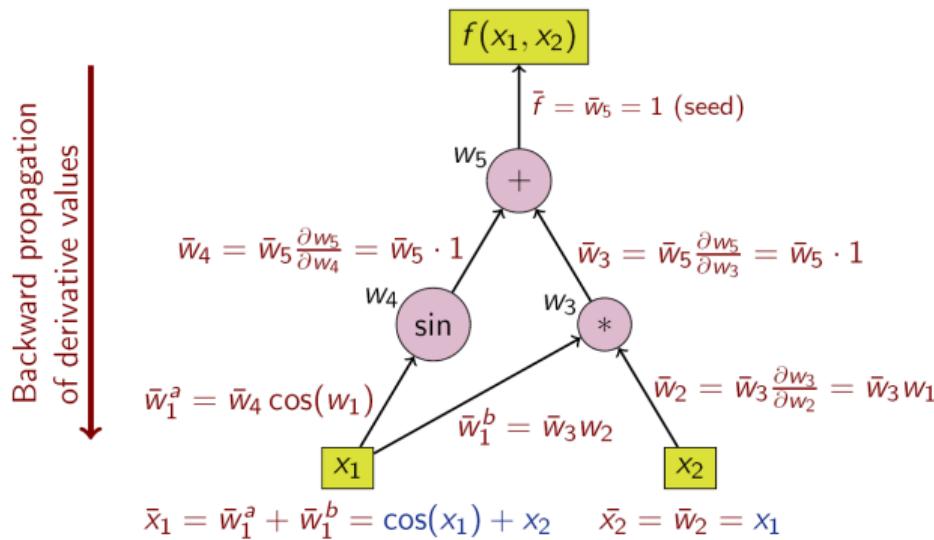
Gradient automatique

- Graphe computationnel : représenter les opérations mathématiques d'un réseau dans graphe
 - Capture l'ordre et la nature des opérations
- Gradient automatique : calculer les gradients **analytiques** sur l'ensemble du réseau automatiquement, via les graphes computationnels
- Permet de définir des topologies complexes et hétérogènes de réseau sans devoir faire les dérivées analytiques manuellement !
- Permet également d'optimiser les traitements sur l'architecture visée (ex. GPU)

Gradient automatique

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = \frac{\partial}{\partial x_1} (\sin(x_1) + x_1 x_2) = \cos(x_1) + x_2$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = \frac{\partial}{\partial x_2} (\sin(x_1) + x_1 x_2) = x_1$$



Outils pour l'apprentissage profond

- TensorFlow (Google) : <https://www.tensorflow.org/>
 - Lancé en novembre 2015, adoption massive par la communauté
 - Code en C++, avec interface d'utilisation en Python
 - Entièrement organisé autour de graphes computationnels
- PyTorch (open source) : <https://pytorch.org/>
 - Basé sur Torch (Collobert et collaborateurs), librairie C++ de plus de 15 ans :
<http://torch.ch/>
 - Offre interface de programmation convivial en Python, approche *programmatique*
 - Différentiation automatique faite dynamiquement, plus versatile sur certains aspects que TensorFlow

Références

-  Yann LeCun, Yoshua Bengio et Geoffrey Hinton. *Deep learning*. Nature, vol. 521, pages 436–444, 2015. <https://doi.org/10.1038/nature14539>
-  Ian Goodfellow, Yoshua Bengio et Aaron Courville. “Deep Learning”, MIT Press, 2016. <http://www.deeplearningbook.org/>
-  Geoffrey Hinton, Yoshua Bengio et Yann LeCun, *Deep Learning NIPS'15 Tutorial*, 2015. <https://nips.cc/Conferences/2015/Schedule?showEvent=4891>