UNIVERSITÉ LAVAL

GLO-4030/7030 APPRENTISSAGE PAR RÉSEAUX DE NEURONES PROFONDS

Sujets avancés :
Ladder network,
réseaux siamois
et distillation

Philippe Giguère

Ladder Network

Rasmus et al., Semi-supervised learning with Ladder networks, NIPS 2015.

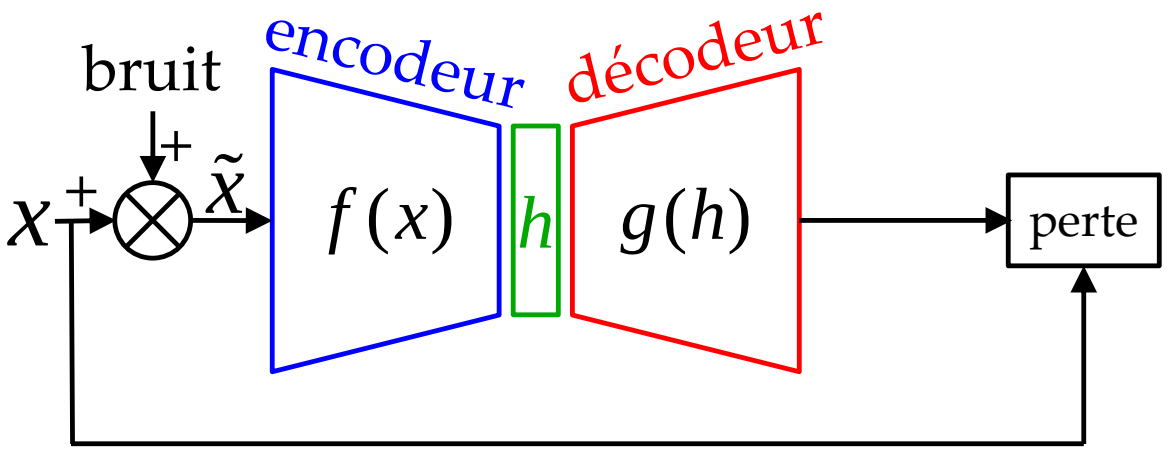
Motivations

- Pas juste un « hack » : fondement théorique sur les liens entre :
  - denoising AE (manifold)
  - modélisation des distributions des variables cachées

    « There is a close connection between denoising and probabilistic modeling »
- Améliore les résultats dans les contextes :
  - entièrement supervisés
  - semi-supervisés
- Raisonnement : perte de reconstruction des DAE ne guide pas toujours le réseau vers des *features* utiles pour classification

Rappel : AE denoising

- Ajoute du bruit aléatoire à l'entrée x

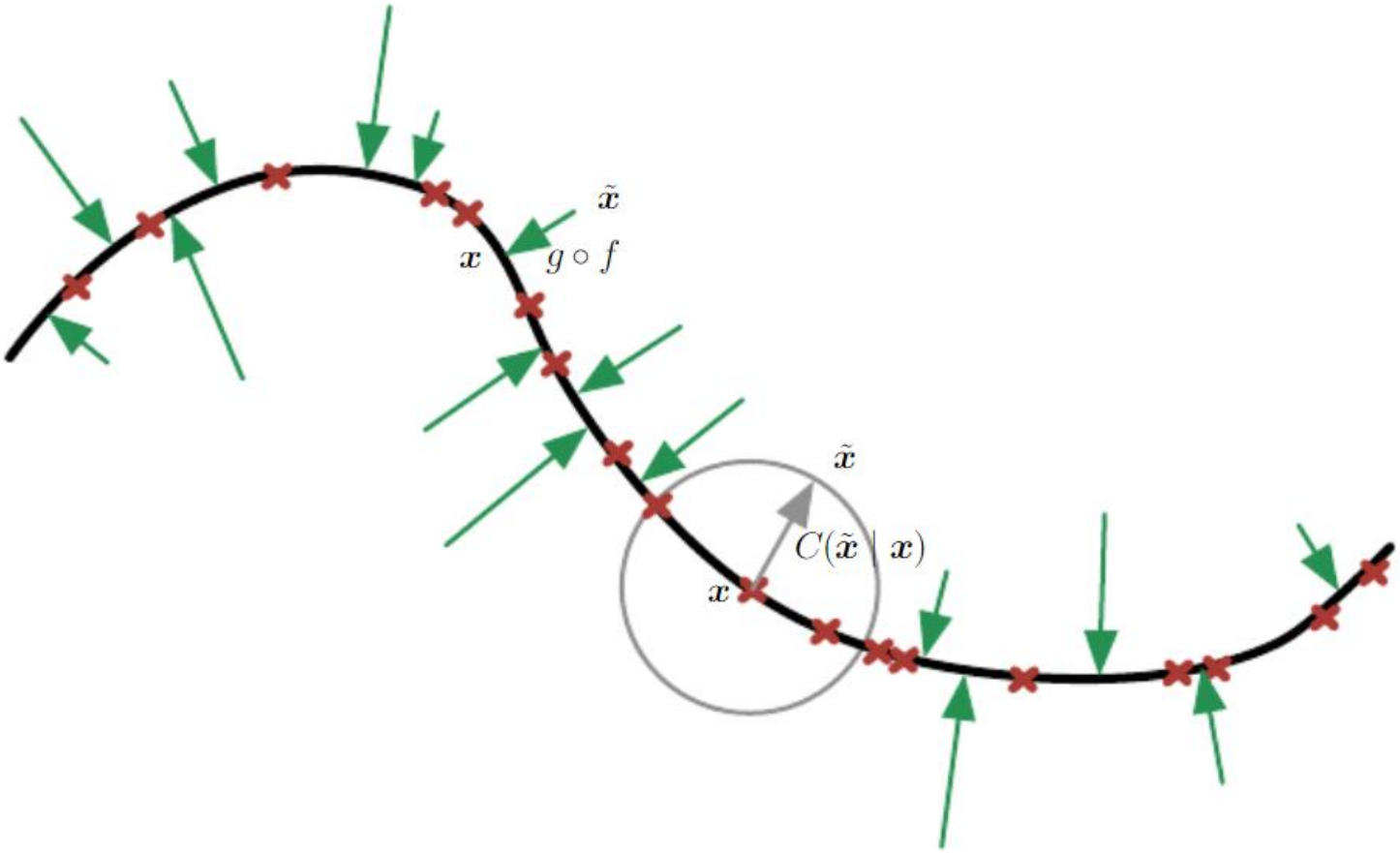


- Cherche quand même à reconstruire x

$$L\left(x, g\left(f\left(\tilde{x}\right)\right)\right)$$

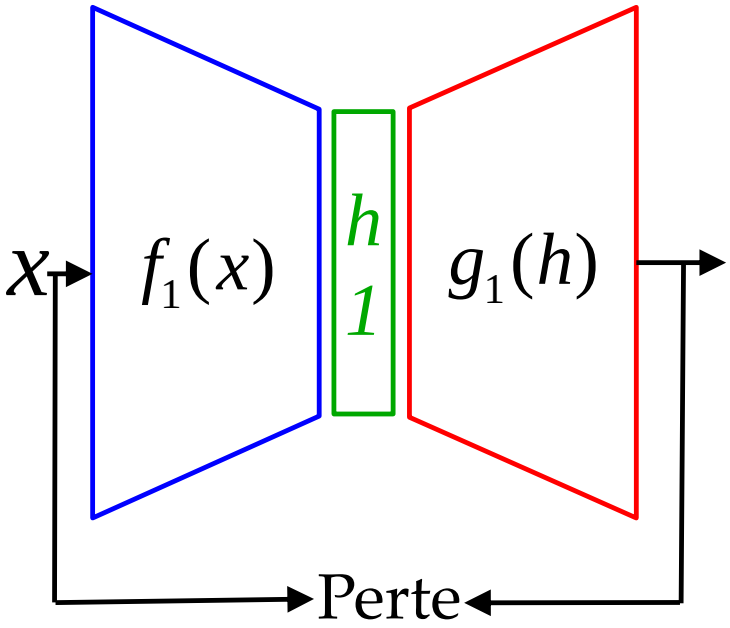
Rappel : AE denoising

- Apprend à déplacer des entrées corrompues $\tilde{x}$ vers le manifold



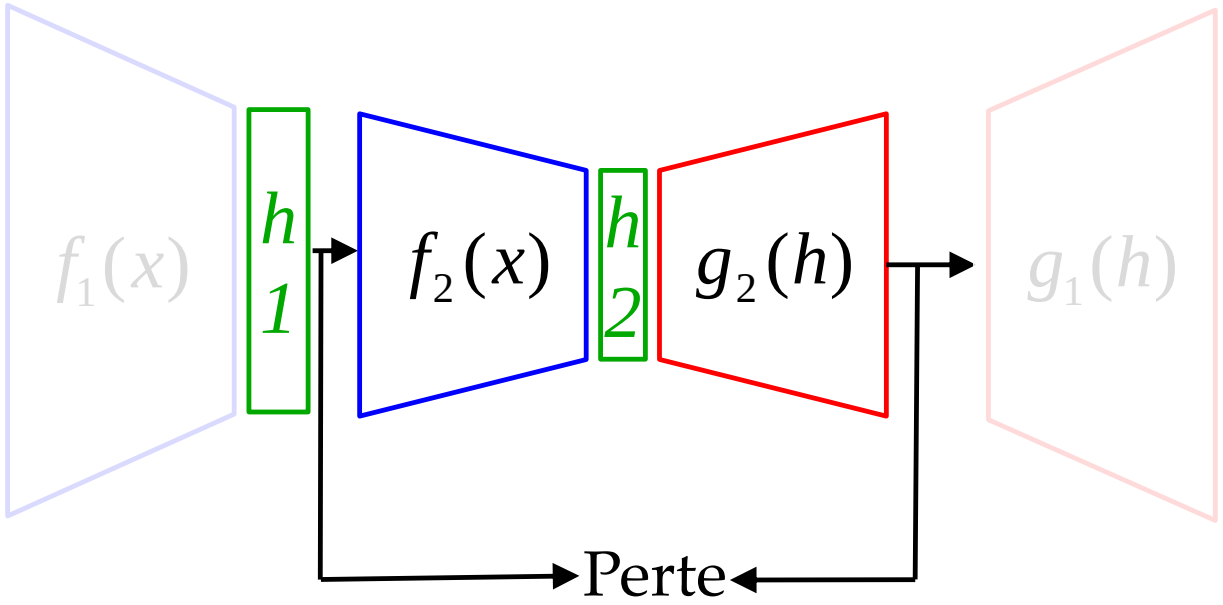
Rappel : Stacked/Deep AE

- Entraînement vorace couche par couche

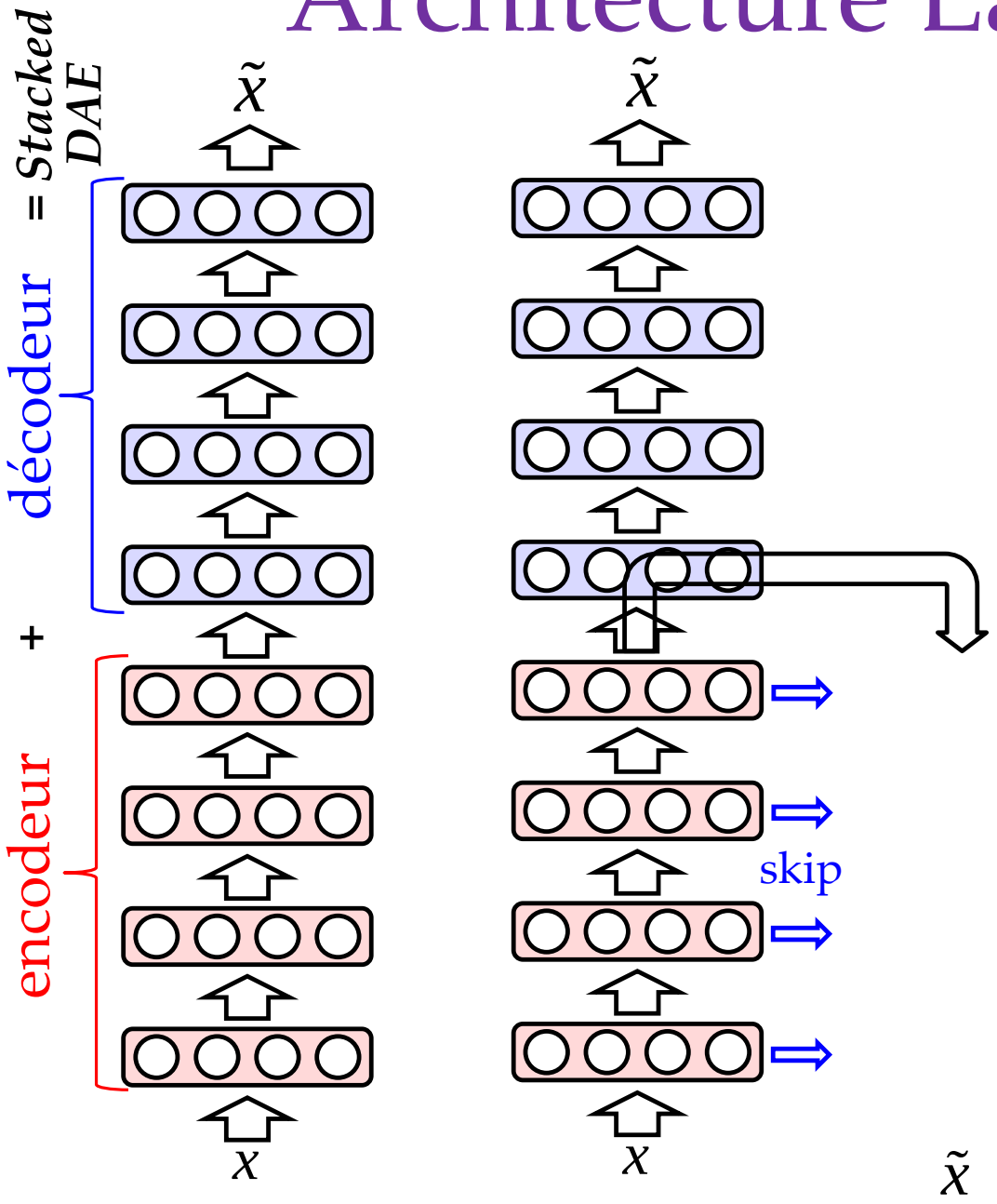


Rappel : Stacked/Deep AE

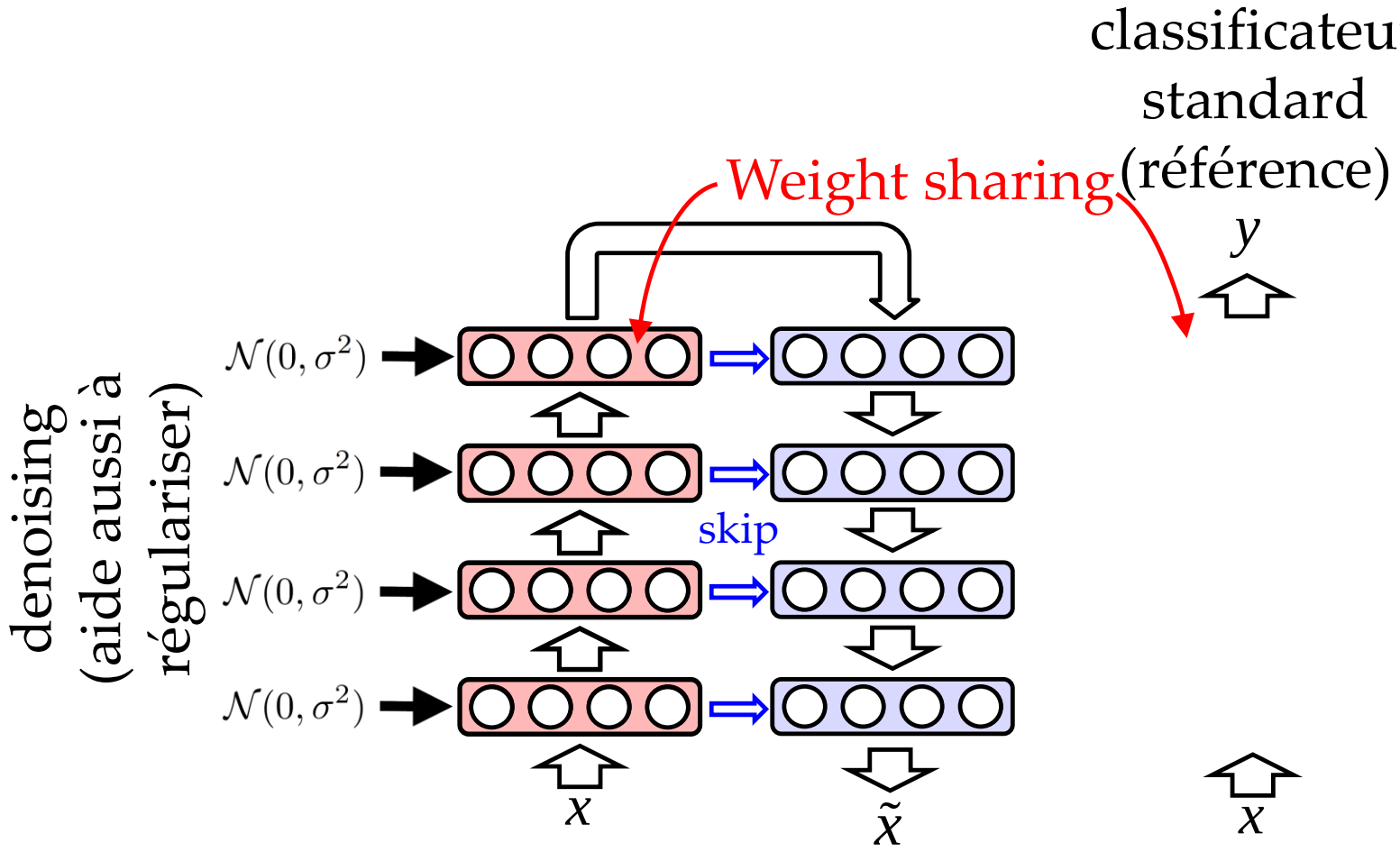
- Entraînement vorace couche par couche



Architecture Ladder Net



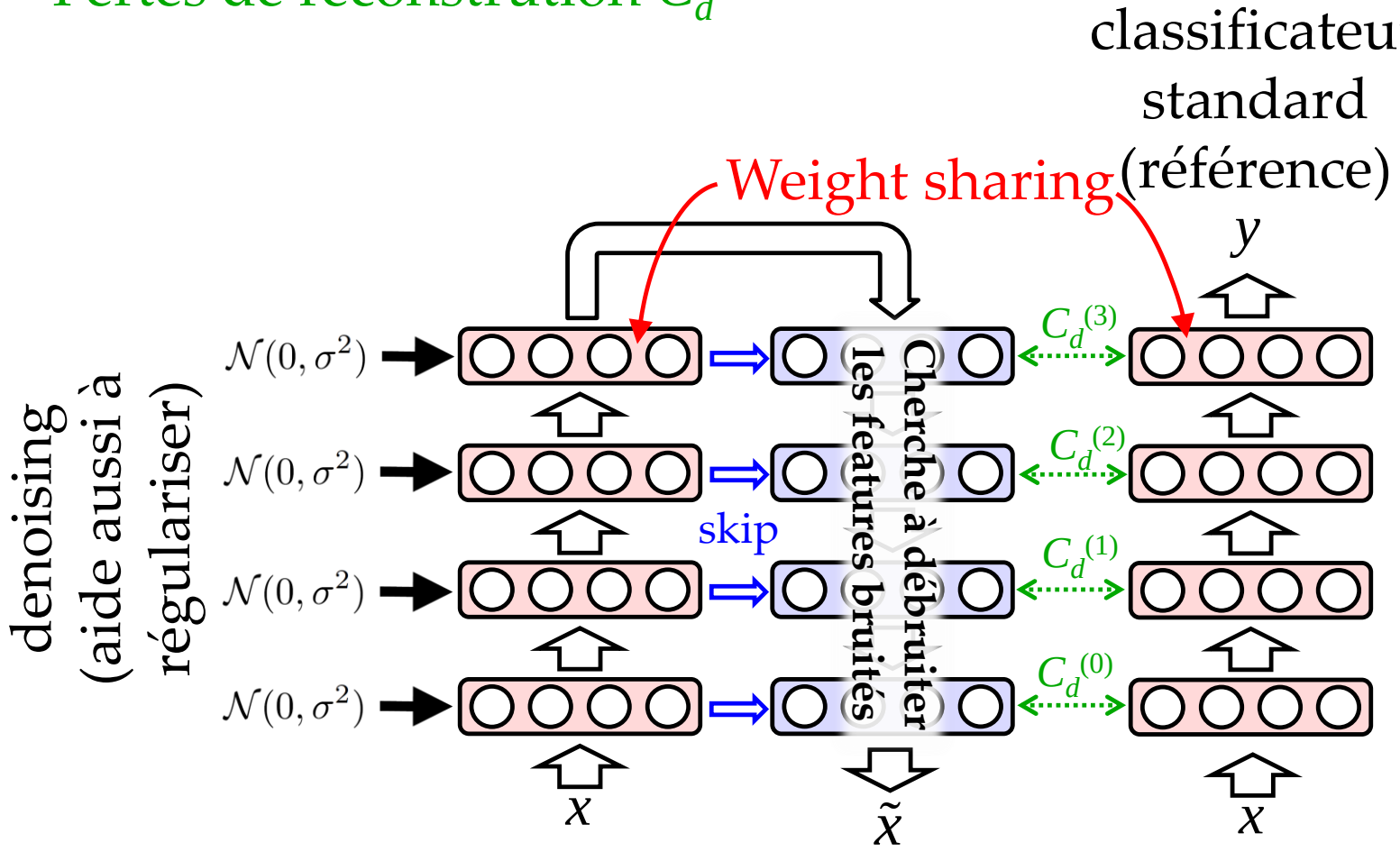
Architecture Ladder Net



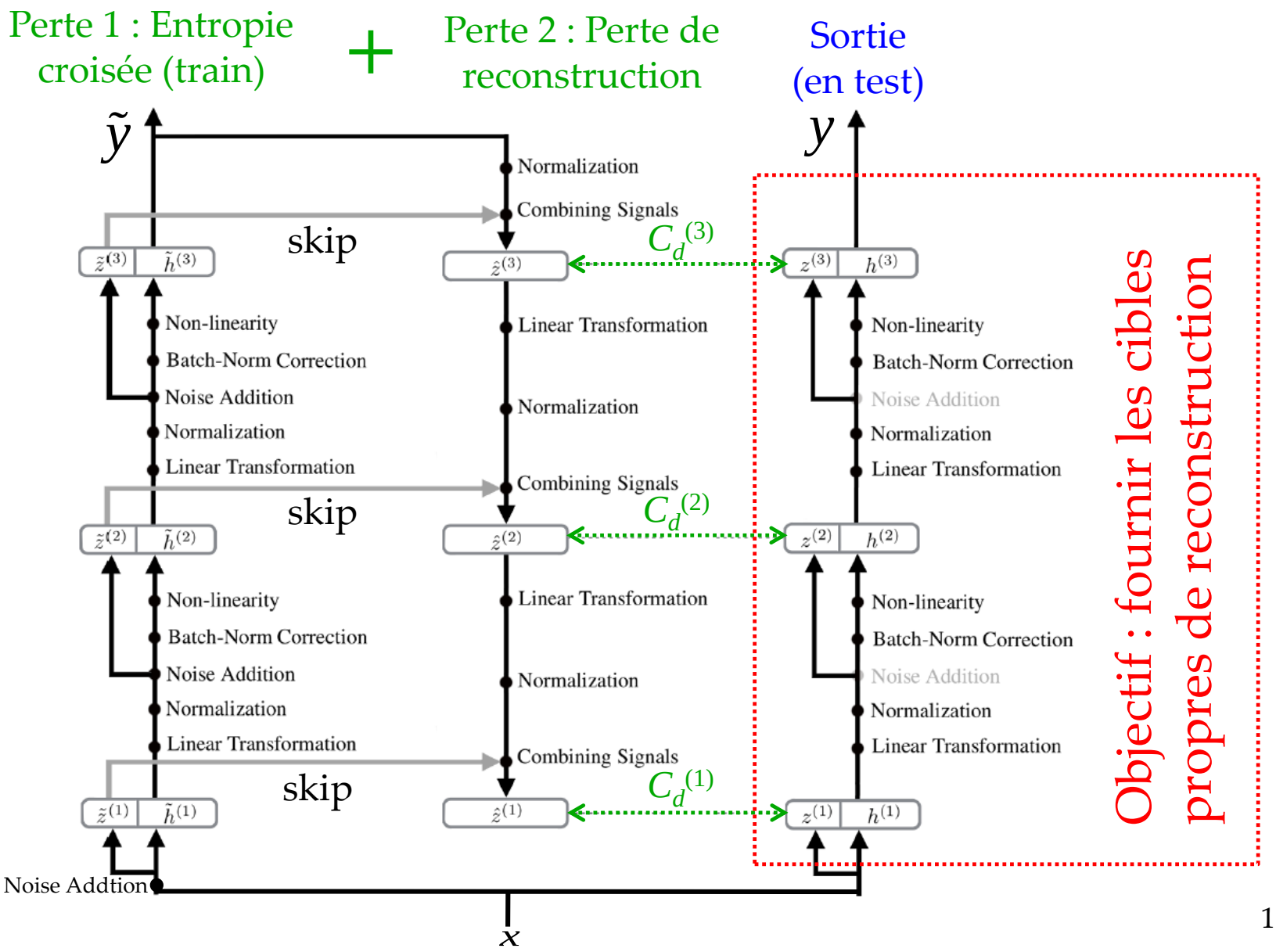
Architecture Ladder Net

Veut que le *denoising* AE apprenne à reconstruire des features utiles à la tâche de classification

Pertes de reconstrution C_d



Architecture : détails



Pezeshki et al., Deconstructing the Ladder Network Architecture, ICML 2016.

2 tâches

- Débruiter (via DAE)
  - apprendre la structure du manifold, distribution des variables latentes
  - via pertes C_d
- Classifier les exemples d'entraînement
  - via perte entropie croisée sur $\tilde{y}$
- Backprop va modifier les poids de l'encodeur/classificateur via ces 2 pertes (weight sharing)
- Au test, classifier avec y

Avantages

- Compatible avec les réseaux *fully-connected* et CNN
- Simple : ajouter un décodeur non-supervisé en tâche auxiliaire de denoising
- Relativement efficace (n'augmente que d'un facteur 3x)

Semi-supervisé avec Ladder

- Peut booster les résultats supervisés avec des données non-étiquetées
- Perte cross-entropie sera ignorée
- Intervertit l'ordre par rapport à faire un pretraining unsupervised + fine-tuning supervisé

Résultats semi-supervisé

Utilise la perte de reconstruction sur décodeur sur tous les exemples, étiquetés ou non.

Permutation invariance means that the model must be unaware of the image (2-D) structure of the data (in other words, CNNs are forbidden). Pas le droit non plus au Data Augmentation par distorsion géométriques

Table 1: A collection of previously reported MNIST test errors in the permutation invariant setting followed by the results with the Ladder network. * = SVM. Standard deviation in parentheses.

Test error % with # of used labels	100	1000	All (50,000)
Semi-sup. Embedding (Weston *et al.*, 2012)	16.86	5.73	1.5
Transductive SVM (from Weston *et al.*, 2012)	16.81	5.38	1.40*
MTC (Rifai *et al.*, 2011)	12.03	3.64	0.81
Pseudo-label (Lee, 2013)	10.49	3.46	
AtlasRBF (Pitelis *et al.*, 2014)	8.10 (± 0.95)	3.68 (± 0.12)	1.31
DGN (Kingma *et al.*, 2014)	3.33 (± 0.14)	2.40 (± 0.02)	0.96
DBM, Dropout (Srivastava *et al.*, 2014)			0.79
Adversarial (Goodfellow *et al.*, 2015)			0.78
Virtual Adversarial (Miyato *et al.*, 2015)	2.12	1.32	0.64 (± 0.03)
Baseline: MLP, BN, Gaussian noise	21.74 (± 1.77)	5.70 (± 0.20)	0.80 (± 0.03)
Γ-model (Ladder with only top-level cost)	3.06 (± 1.44)	1.53 (± 0.10)	0.78 (± 0.03)
Ladder, only bottom-level cost	1.09 (±0.32)	0.90 (± 0.05)	0.59 (± 0.03)
Ladder, full	**1.06** (± 0.37)	**0.84** (± 0.08)	**0.57** (± 0.02)

⇩ compétiteurs

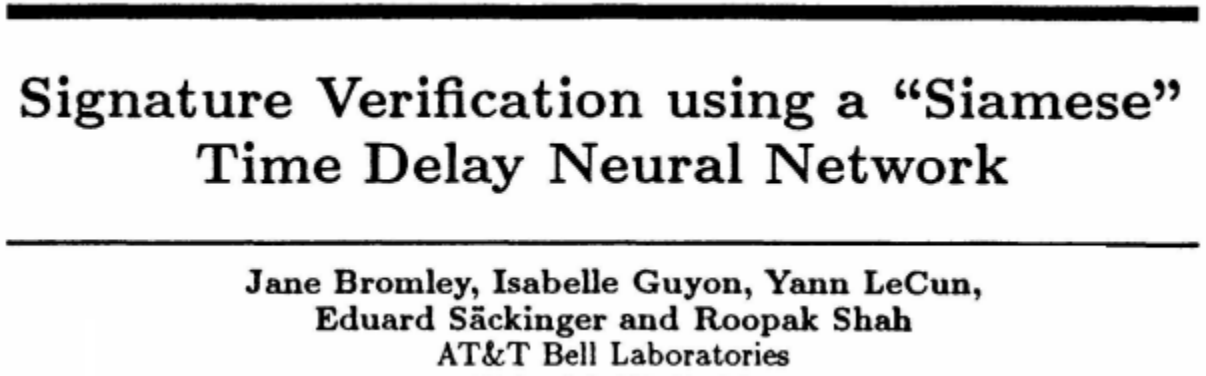
Performe très bien avec seulement 100 exemples étiquetés

A battu l'état de l'art supervisé

Réseau siamois

Réseaux siamois

- Utilisés pour faire des mesures de similarité entre des données
  - comparer des signatures manuscrites
  - est-ce le visage de la même personne
- Introduits en 1993

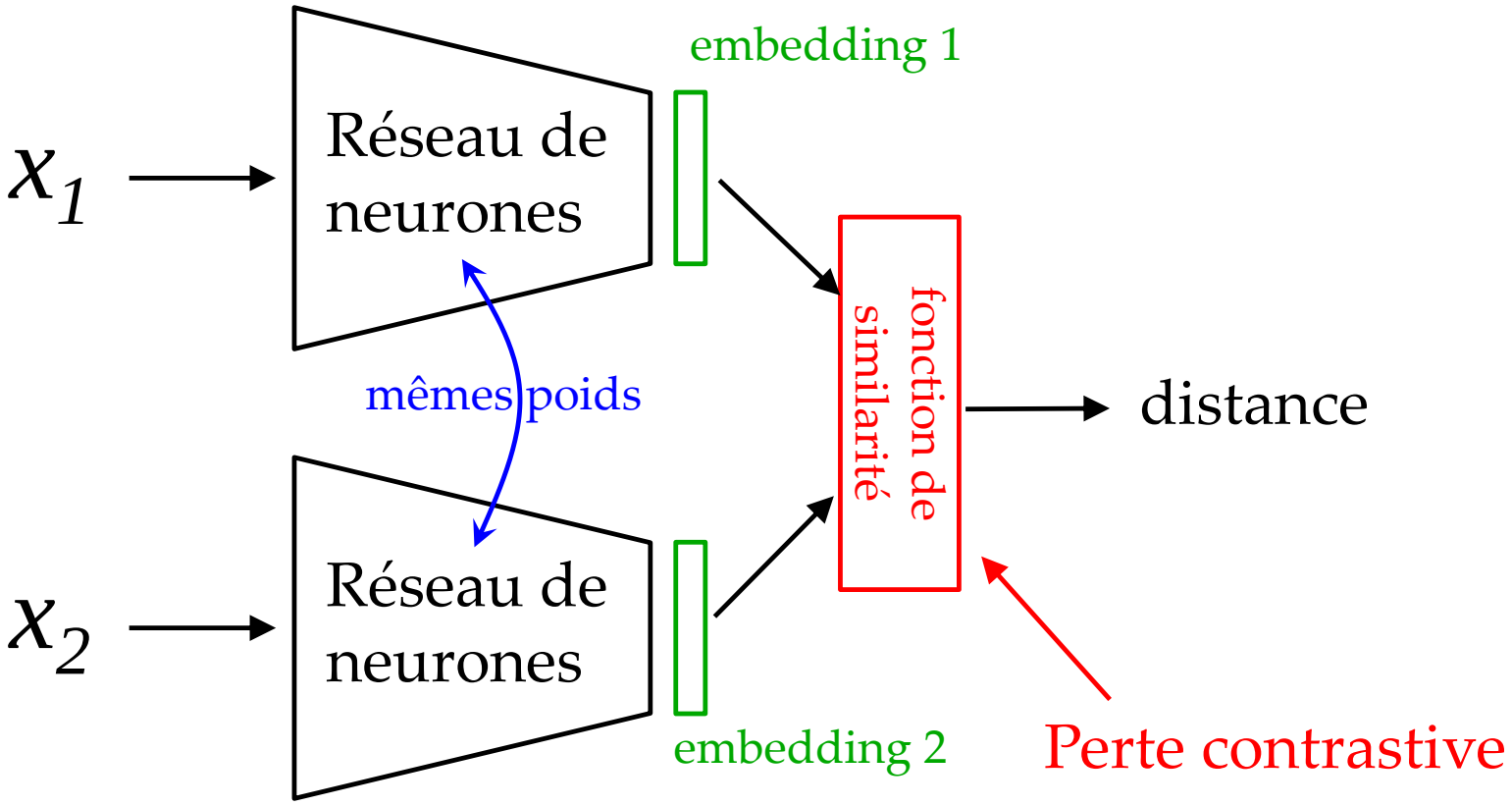


Signature Verification using a "Siamese" Time Delay Neural Network

Jane Bromley, Isabelle Guyon, Yann LeCun,
Eduard Säckinger and Roopak Shah
AT&T Bell Laboratories

- L'entraînement servira à trouver des réseaux générant des embeddings qui performent bien pour ces tâches

Réseaux siamois



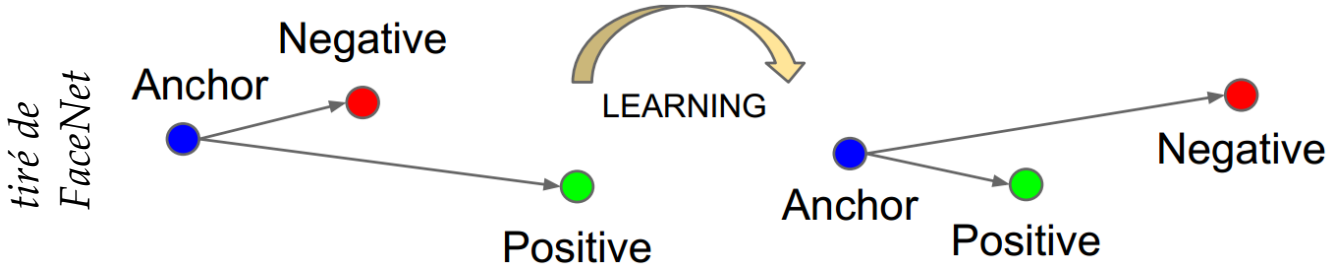
- Trouver un réseau qui :
  - diminue la distance si c'est la même « classe »
  - augmente la distance si c'est des « classes » différentes

Perte triple

- $f(\bullet)$: réseau
- A : anchor P : exemple positif N : exemple négatif



- Cherche à accomplir :



- α : marge, pour éviter la sortie triviale $f(\bullet)=0$

$$\|f(A)-f(P)\|^2+\alpha \le \|f(A)-f(N)\|^2$$

$$L(A,P,N)=\max(\|f(A)-f(P)\|^2-\|f(A)-f(N)\|^2+\alpha,0)$$

"This objective is combined with standard backpropagation algorithm, where the gradient is additive across the twin networks due to the tied weights."

Choix des triplets (FaceNet)

- Si on choisit A, P, et N au hasard, la perte sera généralement nulle
- Doit trouver une collection de triplets qui ont des pertes actives et intéressantes
- Lien avec *Hard negative mining*
- Choisir des triplets de difficultés croissantes, au fil de l'entrainement du réseau : *Curriculum learning, Bengio et al., ICML 2009*

  1) Cleaner examples may yield better generalization faster

  2) Introducing gradually more difficult examples speeds-up online training

Choix des triplets (FaceNet)

- Commence par des exemples négatifs *semi-hard* :

$$\|f(A) - f(P)\|^2 < \|f(A) - f(N)\|^2$$

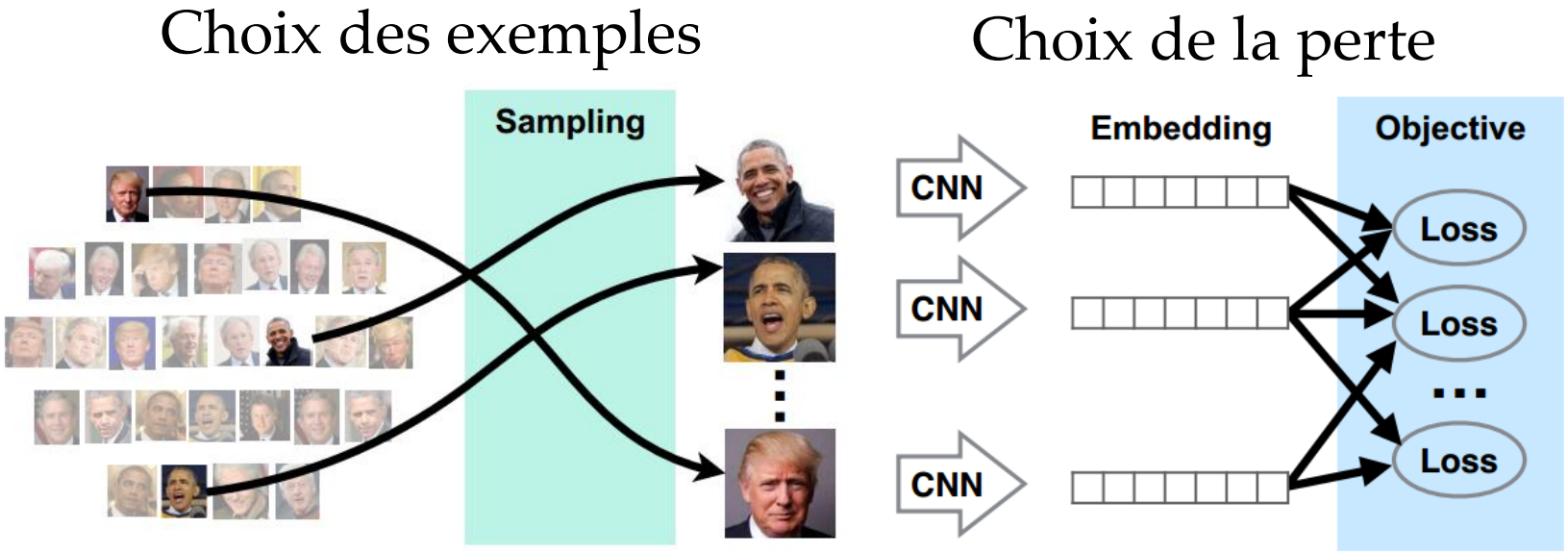
- Migrer vers des *hard-negative*

$$\|f(A) - f(P)\|^2 + \alpha \leq \|f(A) - f(N)\|^2$$

Florian Schroff et al., FaceNet: A Unified Embedding for Face Recognition and Clustering, CVPR 2015.

Sampling Matters in Deep Embedding Learning

- Choix des exemples d'entraînement aussi important que le choix de la perte



- Propose de piger les échantillons en fonction de la distance dans les embeddings

Chao-Yuan Wu et al., Sampling Matters in Deep Embedding Learning, ICCV 2017.

Discussion des pertes

- Perte contrastive pour une paire :

$$L(i,j) = \begin{cases} D_{ij}^2 & \text{si } i, j \text{ sont de la même classe} \\ \max(0, \alpha - D_{ij}^2) & \text{sinon} \end{cases}$$

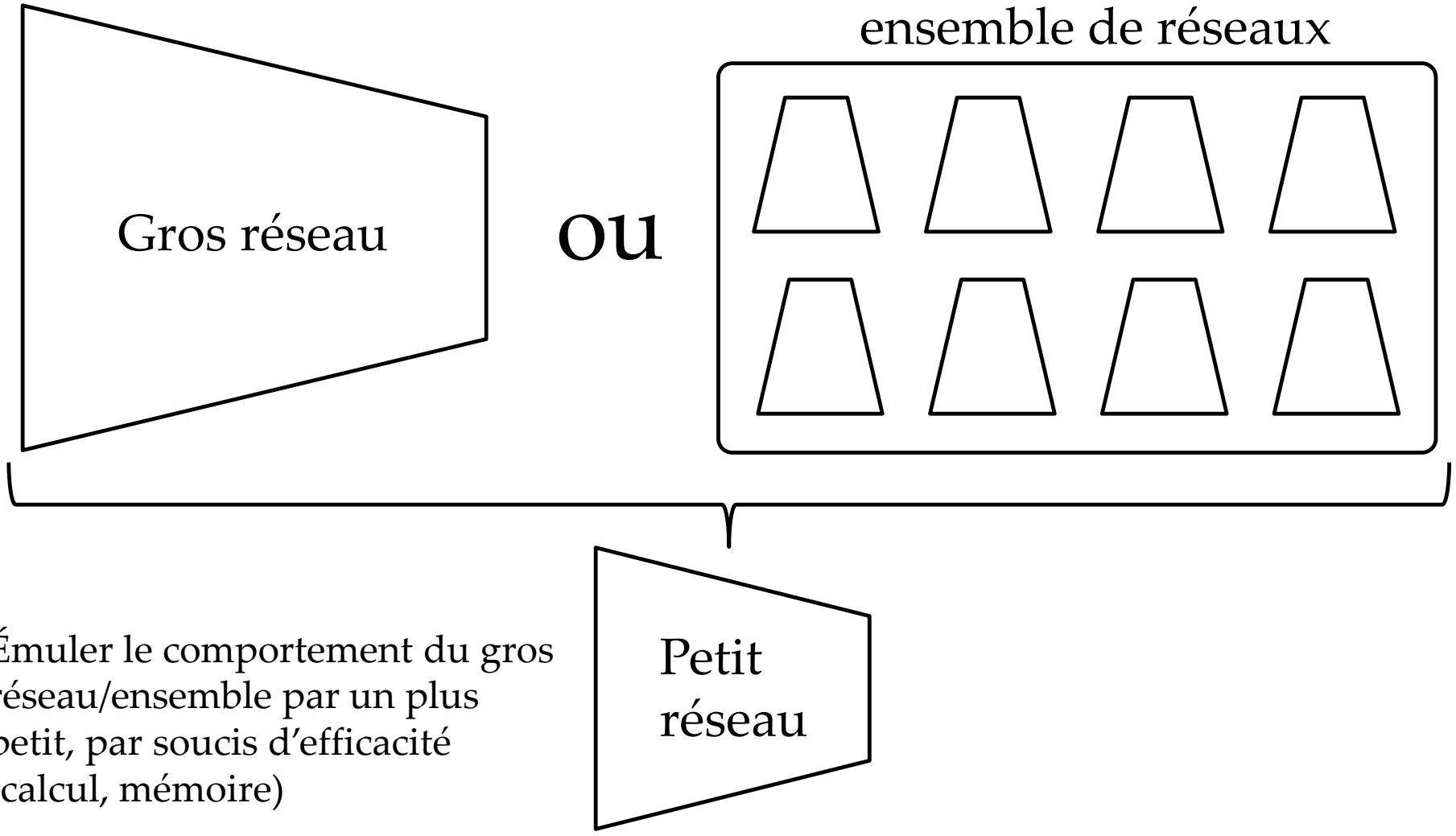
- Tend à pousser :
  - les paires positives (même classe) vers le même embedding
  - les paires négatives vers une distance fixe α
- Perte triple tente plutôt de garder les exemples positifs plus proche entre eux que les négatifs

$$L(A,P,N) = \max(\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha, 0)$$

Chao-Yuan Wu et al., Sampling Matters in Deep Embedding Learning, ICCV 2017.

Distillation

Distillation



Émuler le comportement du gros réseau/ensemble par un plus petit, par soucis d'efficacité (calcul, mémoire)

Distillation

- Apprentissage en deux phases
  - Réseau large/ensemble peut avoir des bonnes caractéristiques de convergences
  - Transfert la ***connaissance*** vers un plus petit
- Connaissance acquise
  - les poids θ d'un réseau se transfert difficilement pour des architectures différentes
    - transfert de connaissance entre cerveaux humains ?
  - voir plutôt comme un *mapping* d'un vecteur x à un vecteur y
  - si le réseau d'origine généralise bien, espoir que le plus petit le fasse aussi

Rappel : Softmax avec température T

$$\hat{y}_i = \frac{\exp(z_i / T)}{\sum_j \exp(z_j / T)}$$

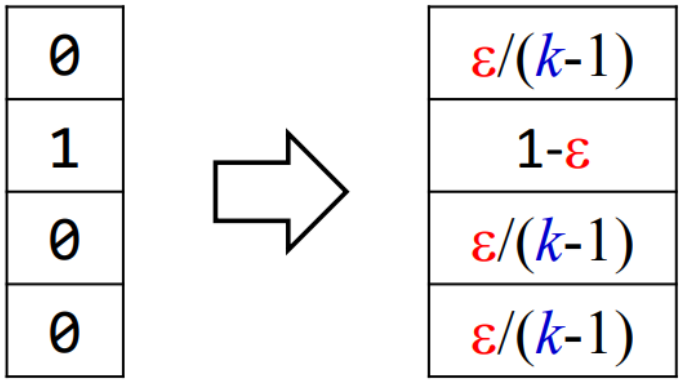
z_i et z_j : scores juste avant la softmax (logit)

- Si T est élevé, les sorties seront plus égalitaires. Si T faible, winner-takes-all
- Va faire ressortir les degrés de similarités entre les classes
  - information supplémentaire

Distillation vs. label smoothing

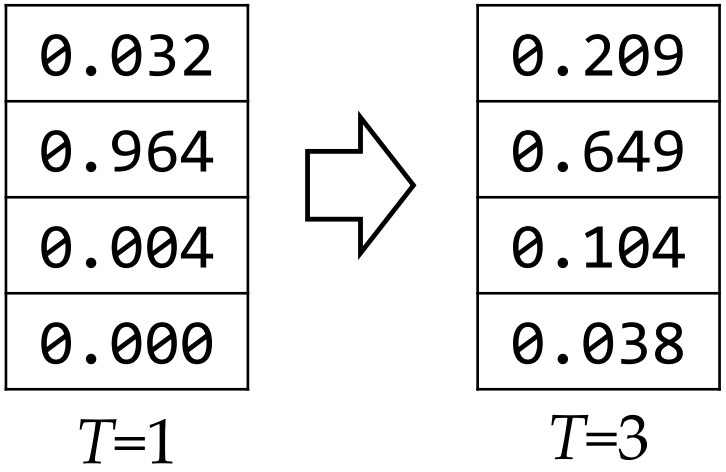
Label smoothing

Donne une distribution uniforme aux autres labels



Softmax + température

Fait ressortir les sorties compétitrices



Exploiter cette information supplémentaire

Données entrainement petit réseau

- Entraîne le réseau A sur données originales $\{X,Y\}$ avec T =1
- Utilise A pour créer deuxième jeu de données D_{dist}
  - hausser la température (exemple T_{dist}=3)
  - obtient un nouveau vecteur cible $\hat{y}$ pour chaque donnée x
  - $D_{dist} = \{X, \hat{Y}\}$
- Entraine le petit réseau B sur D_{dist}
  - utilise T=T_{dist} durant entraînement
  - Perte sur $\hat{Y}$, parfois combinée avec perte sur Y pour T=1
    - comme si on avait deux sorties au réseaux
  - Après entrainement, T=1

(autres détails d'entraînement dans le papier)

Résultats MNIST

- MNIST
  - Réseau A : MLP 2 couches, 1200 unités par c.
    - dropout, weight regularisation, data augmentation par gigue ± 2 pix
    - T=20 pour générer D_{dist}
  - Réseau B : MLP 2 couches, 800 unités par c.
    - pas de régularisation

Réseau	**nombre d'erreurs (test)**
A	67
B (sans distillation)	146
B (**avec distillation**)	74

Résultats JFT

- JFT (Données internes à Google)
  - 100 millions d'images, 15 000 classes
- Réseau baseline
  - 6 mois d'entraînement chez Google!
  - raffinement avec
    - 1 réseau généraliste
    - 61 spécialistes fine-tuned sur 300 classes chacun

System & training set	Train Frame Accuracy	Test Frame Accuracy
Baseline (100% of training set)	63.4%	58.9%
Baseline (3% of training set)	67.3%	44.5%
Soft Targets (3% of training set)	65.4%	57.0%

Distillation semble transférer la capacité à généraliser