

Deep Learning

Caio Corro — Michele Sebag
CNRS — INRIA — LIMSI — LRI

Orsay — Oct. 2019

Credit for slides: Sanjeev Arora; Yoshua Bengio; Yann LeCun; Nando de Freitas; Pascal Germain; Léon Gatys; Weidi Xie; Max Welling; Victor Berger; Kevin Frans; Lars Mescheder et al.; Mehdi Sajjadi et al.



Representation is everything

Auto-Encoders

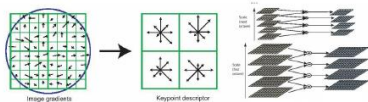
- Denoising Auto-Encoders

- Exploiting latent representations

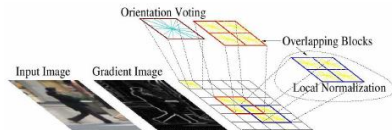
Siamese Networks

The Deep ML revolution: what is new ?

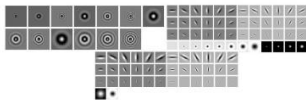
Former state of the art



SIFT



HoG



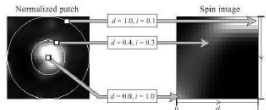
Textons

SIFT: scale invariant feature transform

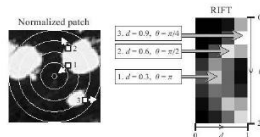
HOG: histogram of oriented gradients

Textons: “vector quantized responses of a linear filter bank”

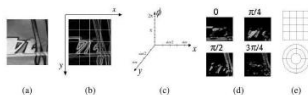
e.g. in computer vision



Spin image



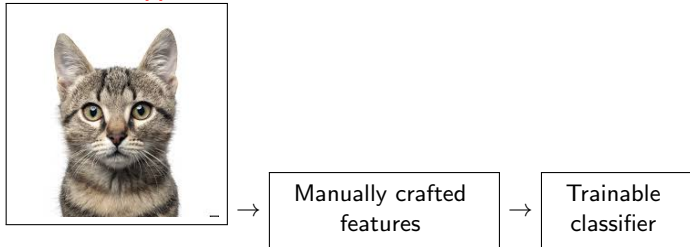
RIFT



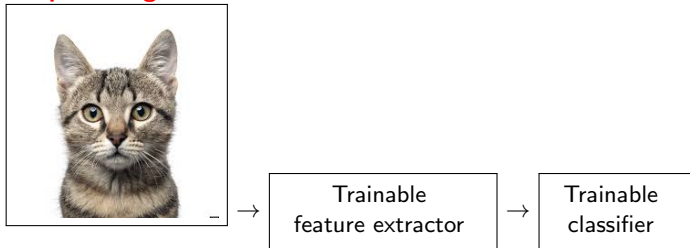
GLOH

What is new, 2

Traditional approach

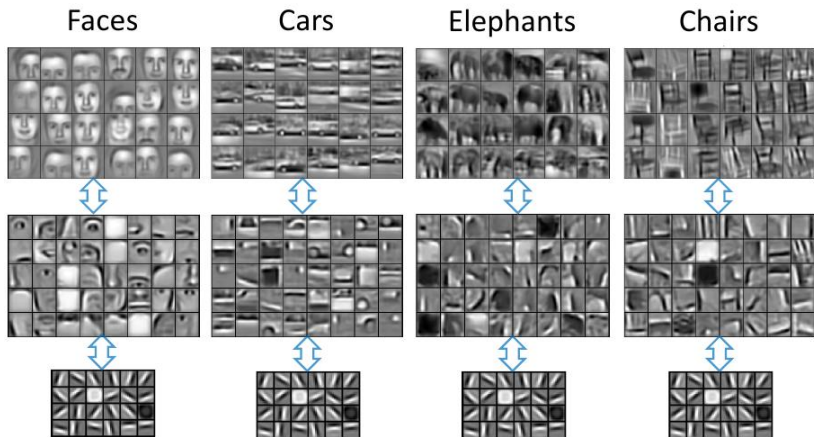


Deep learning



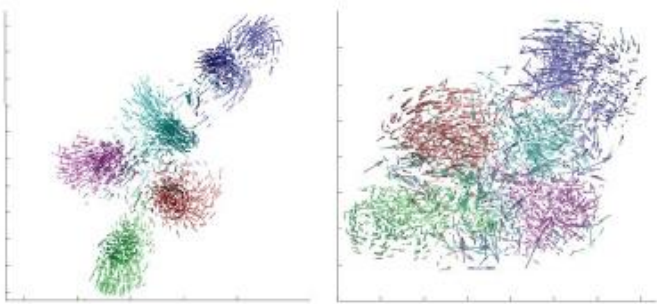
A new representation is learned

Bengio et al. 2006



Good features ?

Le Cun, 2015: <https://www.youtube.com/watch?v=Y-XTcTusUxQ>



Losses and labels



RL (cherry)

SL (icing)

UL (cake)

Lesson learned, quasi consensus @ ICML 2018

- ▶ Introduce many auxiliary tasks / losses
- ▶ Why ? Smoothens the optimization landscape

(conjecture)

<http://unsupervised.cs.princeton.edu/deeplearningtutorial.html>

Key issues

- ▶ Optimization: highly non convex
- ▶ Overparametrisation / Generalization

$$\text{Min } (w - 1)^2 + (w + 1)^2 \quad \text{vs} \quad \text{Min } (w_1 - 1)^2 + (w_2 + 1)^2$$

- ▶ Role of depth
more expressivity; filter noise
- ▶ Make it simpler ? One game is to find a solution; another game is to simplify it.
see also Max Welling's invited talk (Youtube)

Representation is everything

Auto-Encoders

- Denoising Auto-Encoders

- Exploiting latent representations

Siamese Networks

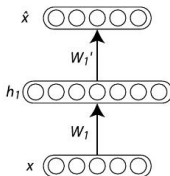
Auto-encoders

$$\mathcal{E} = \{\mathbf{x}_i \in \mathbb{R}^D, i = 1 \dots n\}$$

$$\mathbf{x} \longrightarrow h_1 \in \mathbb{R}^d \longrightarrow \hat{\mathbf{x}}$$

► An auto-encoder:

$$\text{Find } W^* = \arg \min_W \left(\sum_i \|W' \circ W(\mathbf{x}_i) - \mathbf{x}_i\|^2 \right)$$



(*) Instead of min squared error, use cross-entropy loss:

$$\sum_j \mathbf{x}_{i,j} \log \hat{\mathbf{x}}_{i,j} + (1 - \mathbf{x}_{i,j}) \log (1 - \hat{\mathbf{x}}_{i,j})$$

(**) Why W for encoding and W' for decoding ?

Auto-encoders and Principal Component Analysis

Assume

- ▶ A single layer
- ▶ Linear activation

Then

- ▶ An auto-encoder with k hidden neurons \approx first k eigenvectors of PCA

Why ?

Stacked auto-encoders were used to initialize deep networks

In the early Deep Learning era...

Bengio, Lamblin, Popovici, Larochelle 06

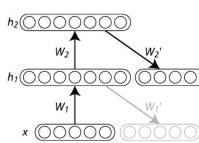
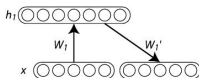
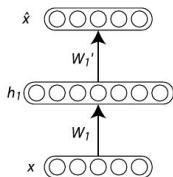
First layer

$$\mathbf{x} \longrightarrow \mathbf{h}_1 \longrightarrow \hat{\mathbf{x}}$$

Second layer

$$\mathbf{h}_1 \longrightarrow \mathbf{h}_2 \longrightarrow \hat{\mathbf{h}}_1$$

same, replacing \mathbf{x} with \mathbf{h}_1



Denoising Auto-Encoders

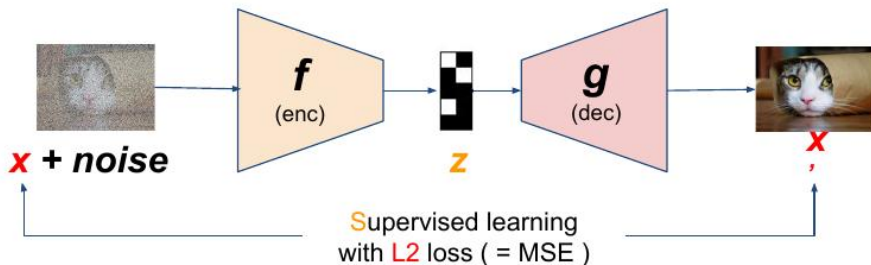
Vincent, Larochelle, Bengio, Manzagol, 08

Principle

- ▶ Add noise to \mathbf{x}
 - ▶ Gaussian noise
 - ▶ Or binary masking noise
- ▶ Recover \mathbf{x} .

(akin drop-out)

$$\text{Find } W^* = \arg \min_W \left(\sum_i \|W' \circ W(\mathbf{x}_i + \epsilon) - \mathbf{x}_i\|^2 \right)$$



Auto-encoders for domain adaptation

Glorot, Bordes, Bengio, 11

Stacked Denoising Auto-Encoders

- ▶ on source and target instances
- ▶ use latent representation to learn on source domain

Why should it work ?

SDAs are able to disentangle hidden factors which explain the variations in the input data, and automatically group features in accordance with their relatedness to these factors. This helps transfer across domains as these generic concepts are invariant to domain-specific vocabularies.

CONS: Computationally expensive

Marginalized Denoising Auto-Encoders

Chen, Xu, Weinberger, Sha 12

Marginalizing a single linear layer

- ▶ \tilde{X} : m copies of X , \tilde{X} : coordinate value independently zeroed with probability p
- ▶ Find $W = \arg \min \| \tilde{X} - W\tilde{X} \|$
- ▶ Solution: $W = PQ^{-1}$ with $P = \tilde{X}\tilde{X}'$ and $Q = \tilde{X}\tilde{X}'$
- ▶ Consider

$$W = \mathbb{E}(P)\mathbb{E}(Q^{-1}) \text{ as } m \rightarrow \infty$$

- ▶ Define $q = (1 - p, \dots, 1 - p, 1)$ (Last entry is for the bias)

$$\mathbb{E}(Q)_{\alpha,\beta} = \begin{cases} X_{\alpha}X'_{\beta}q_{\alpha}q_{\beta} & \text{if } \alpha \neq \beta \\ X_{\alpha}X'_{\alpha}q_{\alpha} & \text{otherwise} \end{cases}$$

Then

- ▶ Inject non-linearity on the top of $W\tilde{X}$ consider $\sigma(W\tilde{X})$
- ▶ Use linear classifier or SVM.

Representation is everything

Auto-Encoders

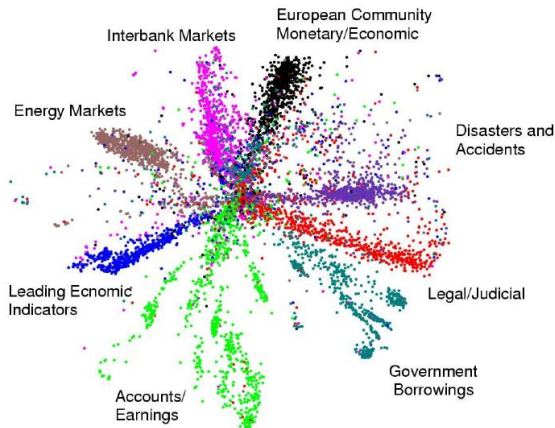
Denoising Auto-Encoders

Exploiting latent representations

Siamese Networks

Visualization with (non linear) Autoencoders

For $d = 2$,



dimensionality reduction on the latent representation

- ▶ Multidimensional scaling
- ▶ t-Distributed Stochastic Neighbor Embedding (t-SNE)

vdMaaten, Hinton 08

t-SNE Do and Don't

<https://distill.pub/2016/misread-tsne/>

Morphing of representations

Gatys et al. 15, 16



Used for *Content*



Used for *Style*

Decrease α/β



- ▶ Style and contents in a convolutional NN are separable
- ▶ Use a trained VGG-19 Net:
 - ▶ applied on image 1 (content)
 - ▶ applied on image 2 (style)
 - ▶ find input matching hidden representation of image 1 (weight α) and hidden representation of image 2 (weight β)

The style

Gatys et al. 15, 16

- Style representation: **correlations** between the different filter responses over the spatial extent of feature maps.
 - Provide colours and local structures.
- Synthesize texture by matching correlation matrices calculated from different layers.
- Key equations: (Check paper for notation)



$$G^l = F^l (F^l)^T$$

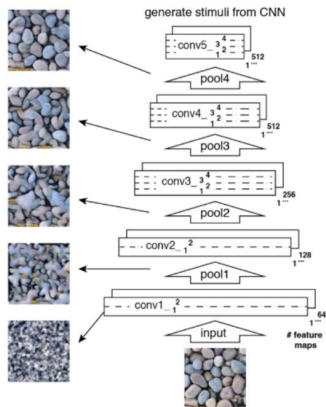
Correlation matrix

$$E_l = \frac{1}{Norm} \sum_{i,j} (G_{ij}^l - A_{i,j}^l)^2$$

Cost for style reconstruction

$$Loss_{style} = \sum_{l=0}^L w_l E_l$$

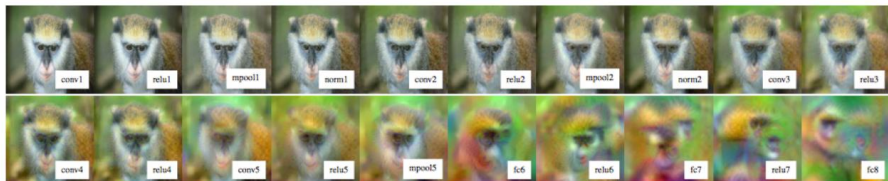
Accumulate cost for lower layers



Portilla & Simoncelli, 2000
Gatys, et al. 2015

The content

Gatys et al. 15, 16



Finally

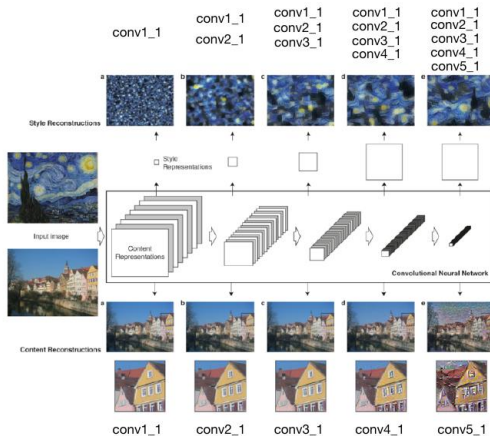
Use image \mathbf{x}_0 for content (AE ϕ), \mathbf{x}_1 for style (AE ϕ')

Morphing: Find input image \mathbf{x} minimizing

$$\alpha \|\phi(\mathbf{x}) - \phi(\mathbf{x}_0)\| + \beta \langle \phi'(\mathbf{x}), \phi'(\mathbf{x}_1) \rangle$$

Morphing of representations, 2

Gatys et al. 15, 16



- ▶ Contents (bottom): convolutions with decreasing precision
- ▶ Style (top): correlations between the convol. features

Morphing of representations, 2

Gatys et al. 15, 16



Representation is everything

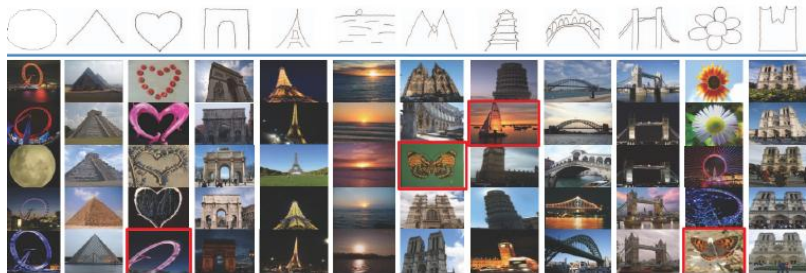
Auto-Encoders

- Denoising Auto-Encoders

- Exploiting latent representations

Siamese Networks

Siamese Networks



Classes or similarities ?

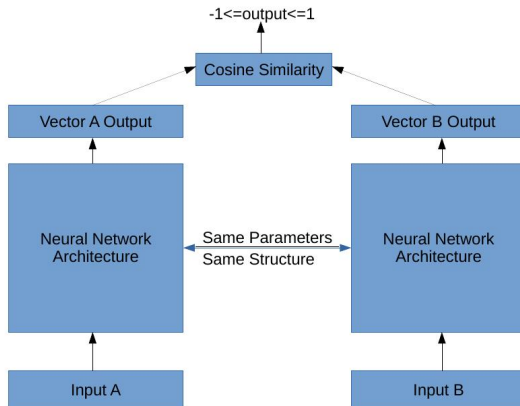
Siamese Networks

Bromley et al. 93

Principle

- ▶ Neural Networks can be used to define a latent representation
- ▶ Siamese: optimize the related metrics

Schema



Siamese Networks, 2

Data

$$\mathcal{E} = \{x_i \in \mathbb{R}^d, i \in [[1, n]]\}; \mathcal{S} = \{(x_{i,\ell}, x_{j_\ell}, c_\ell) \text{ s.t. } c_\ell \in \{-1, 1\}, \ell \in [[1, L]]\}$$

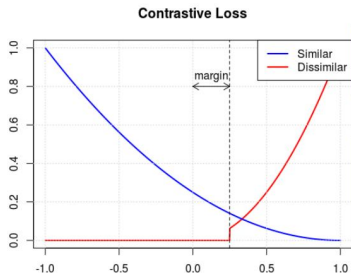
Experimental setting

- ▶ Often: few similar pairs; by default, pairs are dissimilar
- ▶ Subsample dissimilar pairs (optimal ratio between 2/1 ou 10/1)
- ▶ Possible to use domain knowledge in selection of dissimilar pairs

Loss

Given similar and dissimilar pairs (E_+ and E_-)

$$\mathcal{L} = \sum_{(i,j) \in E_+} L_+(i,j) + \sum_{(k,\ell) \in E_-} L_-(k,\ell)$$



$$L_+ = \frac{1}{4} \cdot (1 - \text{cosine}(x_1, x_2))^2$$

$$L_- = \begin{cases} \text{cosine}(x_1, x_2)^2, & \text{if } x \geq \text{margin} \\ 0, & \text{otherwise} \end{cases}$$





Applications

- ▶ Signature recognition
- ▶ Image recognition, search
- ▶ Article, Title
- ▶ Filter out typos
- ▶ Recommendation systems, collaborative filtering

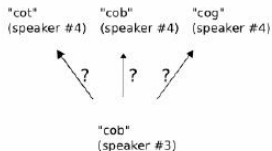
Siamese Networks for one-shot image recognition

Koch, Zemel, Salakhutdinov 15

Training similarity

	same	"cow" (speaker #1)	"cow" (speaker #2)	same
	different	"cow" (speaker #1)	"cat" (speaker #2)	different
	same	"can" (speaker #1)	"can" (speaker #2)	same
	different	"can" (speaker #1)	"cab" (speaker #2)	different

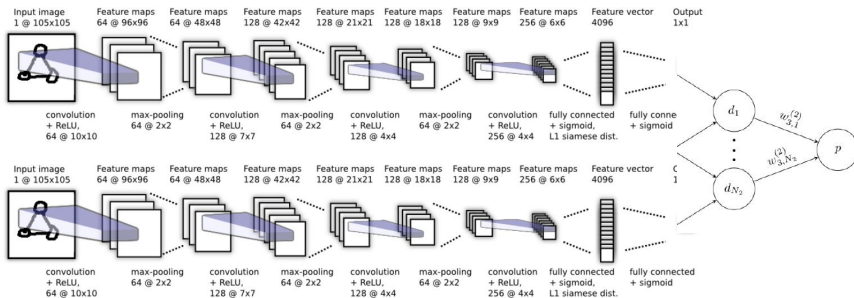
One-shot setting



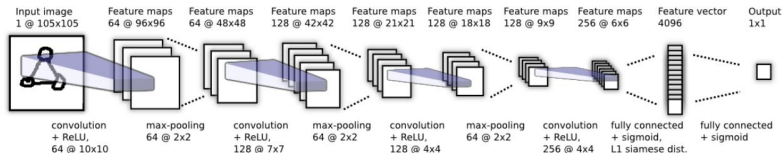
Ingredients

Koch et al. 15

Architecture



Architecture



Distance

$$d(x, x') = \sigma \left(\sum_k \alpha_k |z_k(x) - z_k(x')| \right)$$

Loss

Given a batch $((x_i, x'_i), y_i)$ with $y_i = 1$ iff x_i and x'_i are similar

$$\mathcal{L}(w) = \sum_i y_i \log d(x_i, x'_i) + (1 - y_i) \log (1 - d(x_i, x'_i)) + \lambda \|w\|^2$$

Results

Koch et al. 15

Omniglot



Results

Method	Test
Humans	95.5
Hierarchical Bayesian Program Learning	95.2
Affine model	81.8
Hierarchical Deep	65.2
Deep Boltzmann Machine	62.0
Simple Stroke	35.2
1-Nearest Neighbor	21.7
Siamese Neural Net	58.3
Convolutional Siamese Net	92.0

Siamese Networks

PROS

- ▶ Learn metrics, invariance operators
- ▶ Generalization beyond train data

CONS

- ▶ More computationally intensive
- ▶ More hyperparameters and fine-tuning, more training