INSA Rouen Normandie, Normandie Université
LITIS laboratory ✧ Learning Team

Neural networks regularization through representation learning

PhD defense

Soufiane BELHARBI

soufiane.belharbi@insa-rouen.fr

sbelharbi.github.io

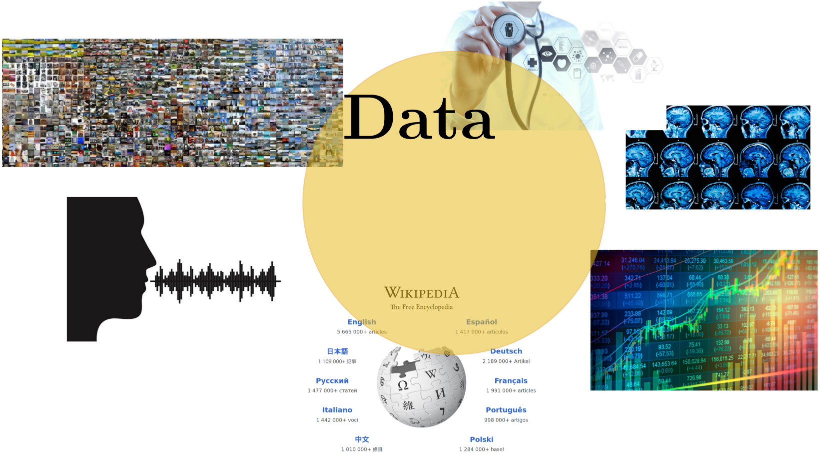
INSA Rouen Normandie

Jury

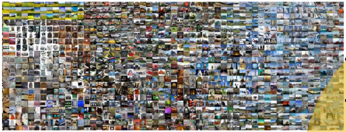
- **Elisa FROMONT**, Professor, Université de Rennes 1 (reporter)
- **Thierry ARTIÈRES**, Professor, École Centrale Marseille (reporter)
- **John LEE**, Professor, Université Catholique de Louvain (examiner)
- **David PICARD**, Assistant professor, École Nationale Supérieure de l'Électronique et de ses Applications (examinateur)
- **Frédéric JURIE**, Professor, Université de Caen Normandie (guest)
- **Sébastien ADAM**, Professor, Université de Rouen Normandie (supervisor)
- **Clément CHATELAIN**, Assistant professor, INSA Rouen Normandie (advisor)
- **Romain HÉRAULT**, Assistant professor, INSA Rouen Normandie (advisor)

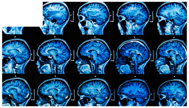
Jul. 06, 2018

Introduction: Learning from data


Data
WIKIPEDIA
The Free Encyclopedia
English
5 665 000+ articles
Español
1 417 000+ artículos
日本語
1 109 000+ 記事
Deutsch
2 189 000+ Artikel
Русский
1 477 000+ статей
Français
1 991 000+ articles
Italiano
1 442 000+ voci
Português
998 000+ artigos
中文
1 010 000+ 條目
Polski
1 284 000+ haseł

Introduction: Learning from data

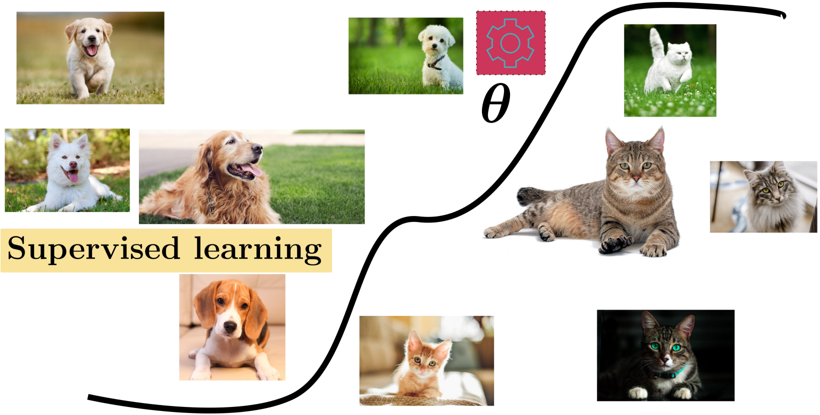




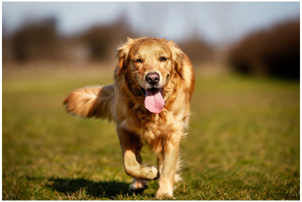
Data

Machine learning:

- ☞ Statistics
- ☞ Knowledge
- ☞ Decision

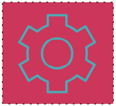

θ
Supervised learning

Prediction: Generalization



What is this?

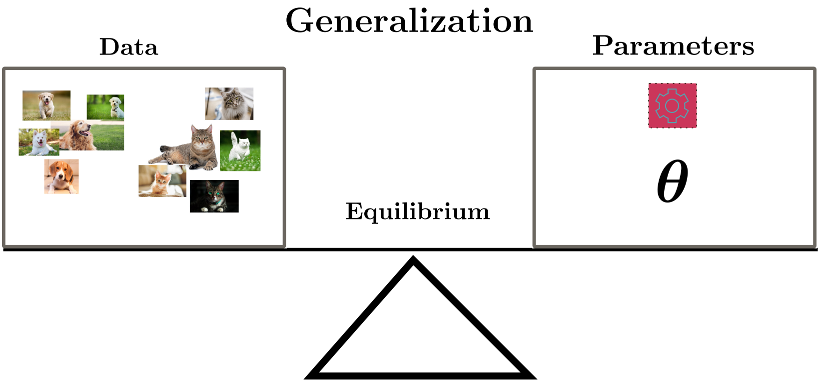
$\longrightarrow$

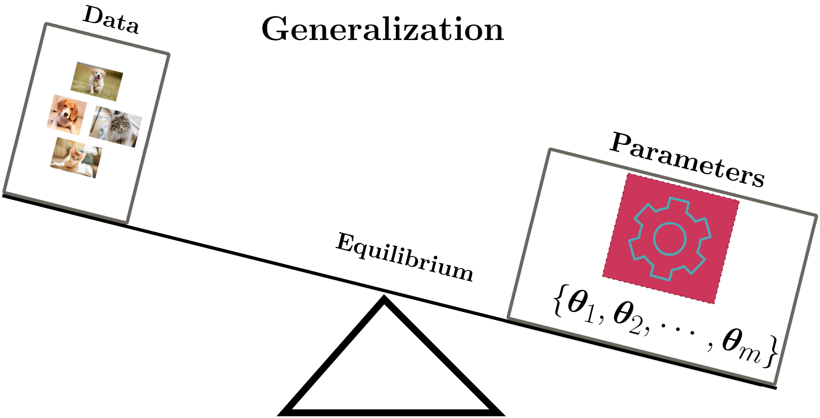


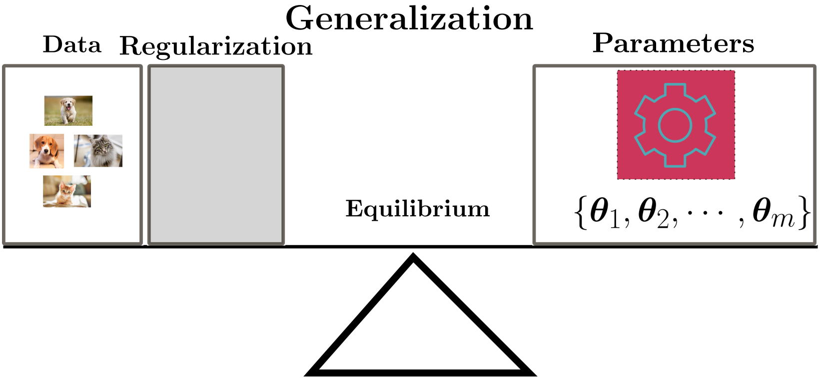
$\boldsymbol{\theta}$

$\longrightarrow$

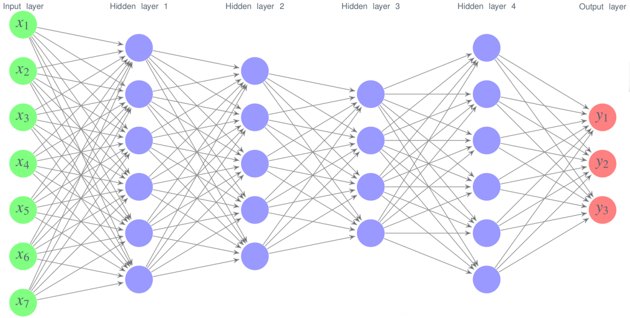
Dog


Generalization
Data
Parameters
θ
Equilibrium


Generalization
Data
Parameters
Equilibrium
$\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \cdots, \boldsymbol{\theta}_m\}$

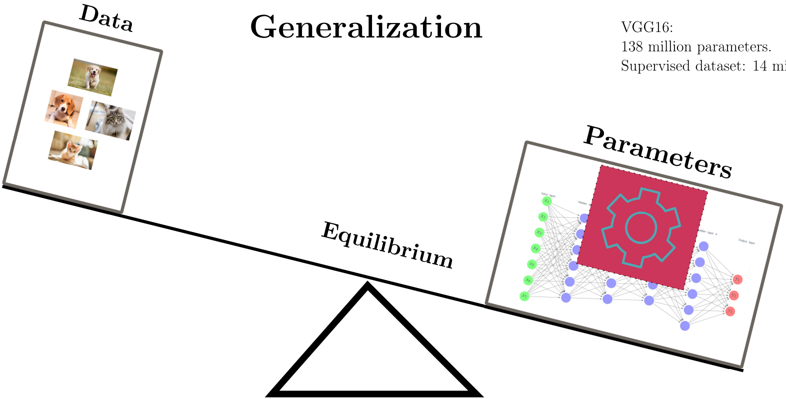

Generalization
Data
Regularization
Parameters
Equilibrium
$\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \cdots, \boldsymbol{\theta}_m\}$

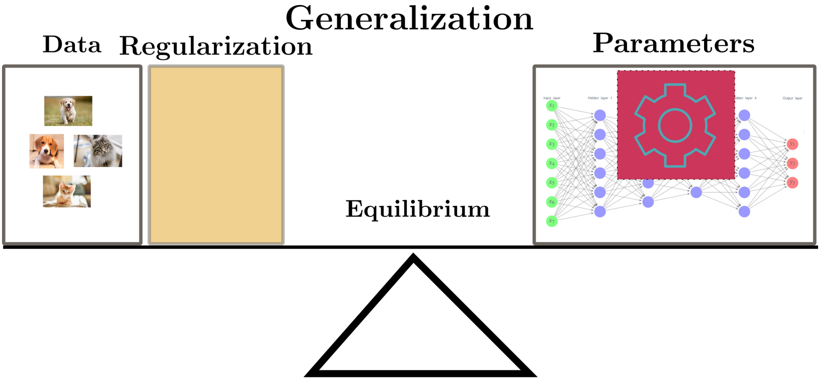
=


Input layer
Hidden layer 1
Hidden layer 2
Hidden layer 3
Hidden layer 4
Output layer
x_1
x_2
x_3
x_4
x_5
x_6
x_7
y_1
y_2
y_3

Generalization

VGG16:
138 million parameters.
Supervised dataset: 14 million.




Generalization
Data
Regularization
Parameters
Equilibrium

Neural networks regularization

Data augmentation

Sparse/invariant representations

Adversarial training

Early stopping

Multi-task learning

L_p parameters norm

Transfer learning

Dropout

Noise injection: data, weights, labels

Parameters sharing

Semi-supervised/unsupervised learning

Batch normalization

Tangent propagation and manifold learning

Neural networks regularization

Data augmentation

Early stopping

L_p parameters norm

Noise injection:
data, weights, labels

Batch normalization

Representation learning

Sparse/invariant representations

Adversarial training

Multi-task learning

Transfer learning

Dropout

Parameters sharing

Semi-supervised/unsupervised learning

Tangent propagation and manifold learning

Neural networks regularization

Data augmentation

Early stopping

L_p parameters norm

Noise injection:
data, weights, labels

Batch normalization

Representation learning

Sparse/invariant representations

Adversarial training

Multi-task learning

Transfer learning

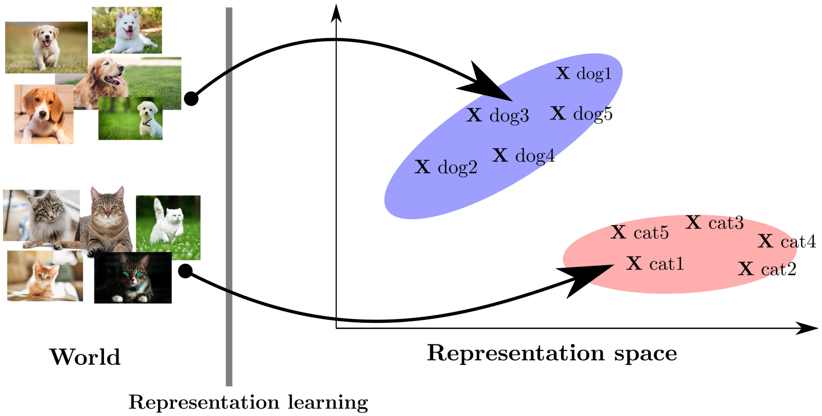
Dropout

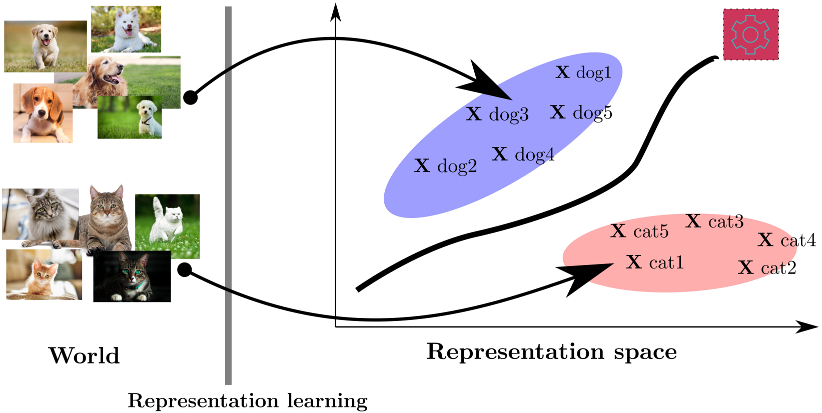
x **This thesis**

Parameters sharing

Semi-supervised/unsupervised learning

Tangent propagation and manifold learning


X dog1
X dog3
X dog5
X dog4
X dog2
X cat5
X cat3
X cat4
X cat1
X cat2
World
Representation space
Representation learning


X dog1
X dog3
X dog5
X dog4
X dog2
X cat5
X cat3
X cat4
X cat1
X cat2
World
Representation space
Representation learning

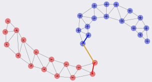
PhD contributions

Contribution 1

- Unsupervised learning: Structured output prediction.

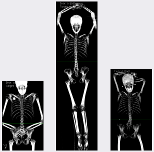
Contribution 2

- Prior knowledge: Classification.



Contribution 3 (medical application)

- Transfer learning: Regression.



Unsupervised learning for structured output predictions

- Traditional Machine Learning Problems $f : \mathcal{X} \rightarrow \mathcal{Y}$
  - Inputs $\mathcal{X} \in \mathbb{R}^d$
  - Outputs $\mathcal{Y} \in \mathbb{R}$ for the task: classification, regression, . . .

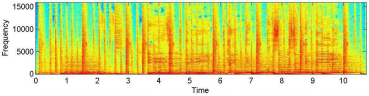
- Machine Learning for *Structured Output* Problems $f : \mathcal{X} \rightarrow \mathcal{Y}$
  - Inputs $\mathcal{X} \in \mathbb{R}^d$
  - Outputs $\mathcal{Y} \in \mathbb{R}^{d'}, d' > 1,$ a structured object: dependencies among its components.

C. Lampert slides.

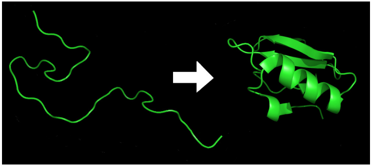
$\textbf{Data} = \textit{representation}\ (\text{values}) + \textit{structure}\ (\text{dependencies})$

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

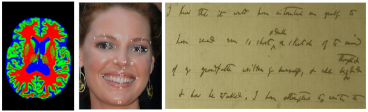
Text: part-of-speech tagging, translation



speech $\rightleftarrows$ text



Protein folding

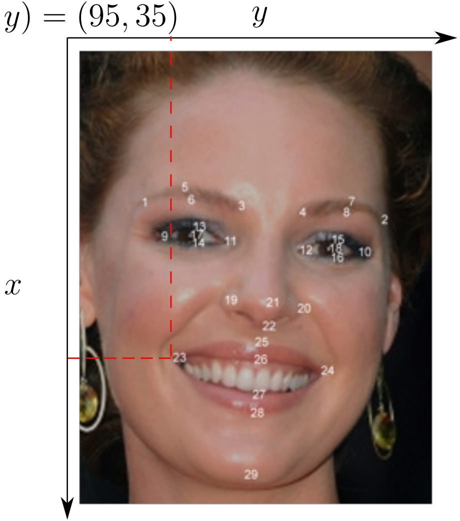


Image

Structured data

Data = *representation* (values) + *structure* (dependencies)

Point 23: $(x, y) = (95, 35)$



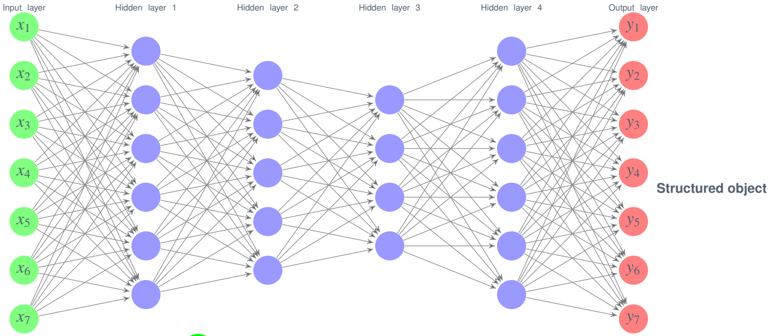
Geometric structured data.

- Approaches that Deal with Structured Output Data:
  - Kernel based methods: Kernel Density Estimation (KDE).
  - Discriminative methods: Structure output SVM.
  - Graphical methods: HMM, CRF, MRF, ...
- Drawbacks:
  - Perform one single data transformation.
  - Most of them have difficulties to deal with *high dimensional* data.
- Ideal approach:
  - High dimension data.
  - Multiple data transformations (complex mapping functions).

- **Neural networks!**

Structured output problems: Feedforward neural networks issue

High dimensional output:



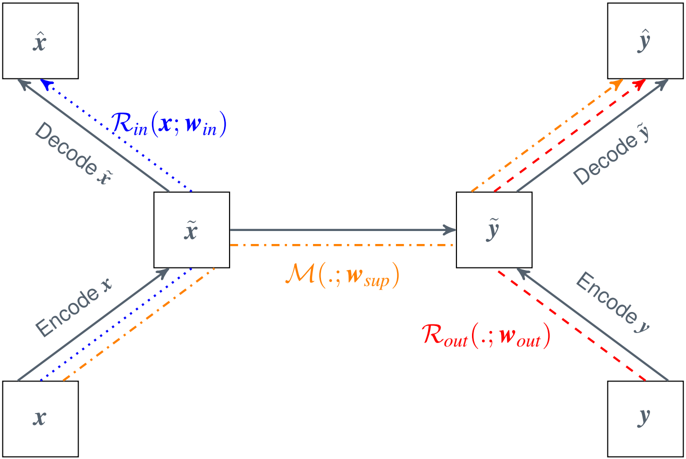
- High dimension data. ☺
- Multiple data transformations (complex mapping functions). ☺

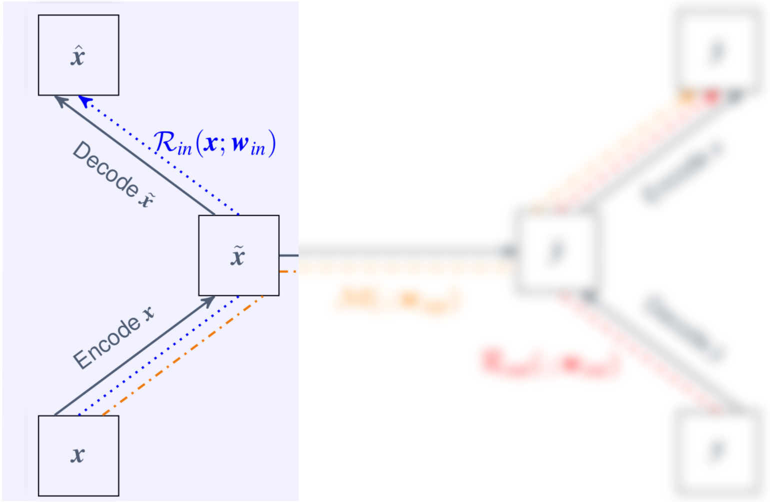
- ⚠ No support to structured output. ☹
- ⚠ Overfitting, output average structure. ☹

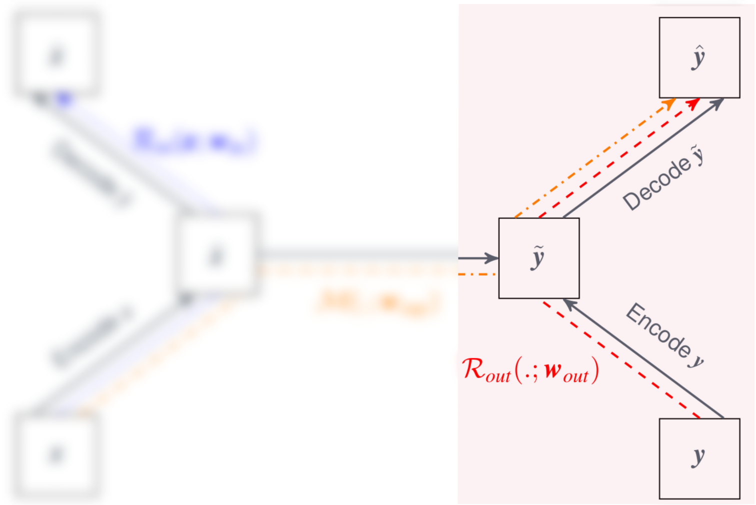
☞ Regularization through unsupervised learning.

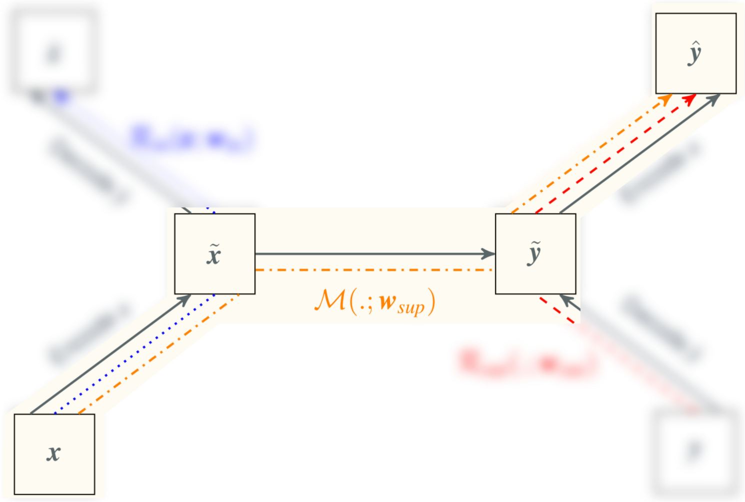
Key idea:

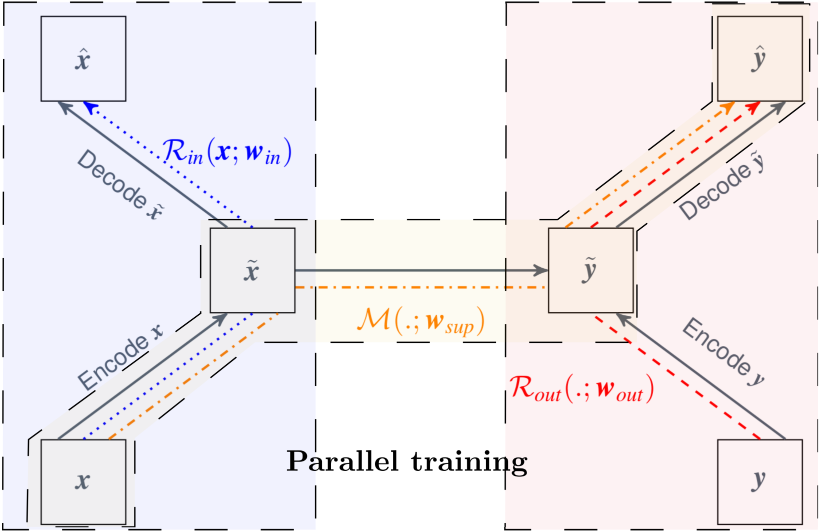
Use unsupervised learning to Learn/discover the hidden structure of the output data.

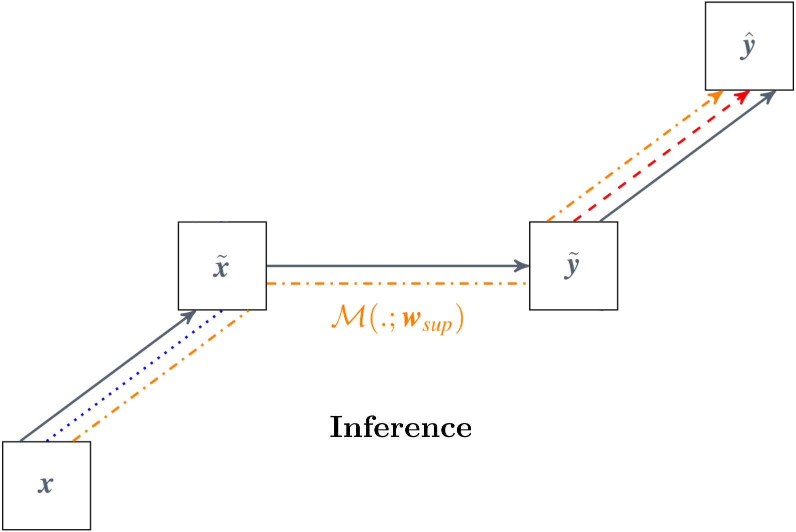

$\hat{x}$
$\hat{y}$
$\mathcal{R}_{in}(\boldsymbol{x};\boldsymbol{w}_{in})$
Decode $\tilde{x}$
Decode $\tilde{y}$
$\tilde{x}$
$\tilde{y}$
$\mathcal{M}(.;\boldsymbol{w}_{sup})$
Encode x
Encode y
$\mathcal{R}_{out}(.;\boldsymbol{w}_{out})$
x
y


$\hat{x}$
$\mathcal{R}_{in}(\boldsymbol{x}; \boldsymbol{w}_{in})$
Decode $\tilde{x}$
$\tilde{x}$
Encode x
x


$\hat{y}$
Decode $\tilde{y}$
$\tilde{y}$
Encode y
$\mathcal{R}_{out}(.; \boldsymbol{w}_{out})$
y

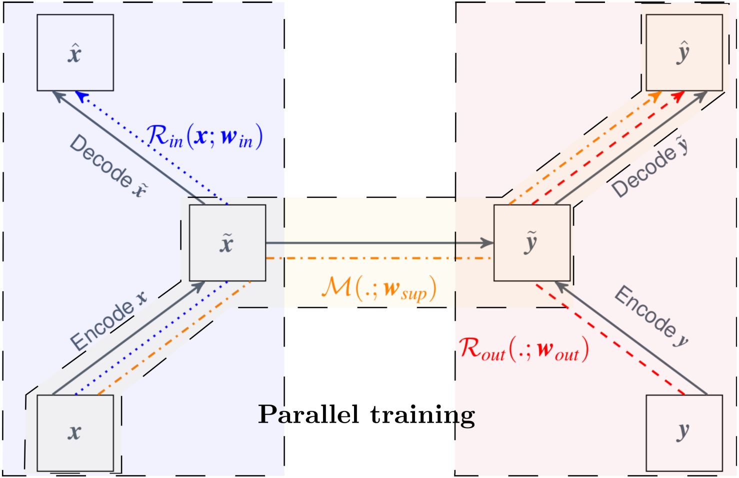

$\hat{y}$
$\tilde{x}$
$\tilde{y}$
$\mathcal{M}(.;\boldsymbol{w}_{sup})$
x


$\hat{x}$
$\mathcal{R}_{in}(\boldsymbol{x}; \boldsymbol{w}_{in})$
Decode $\tilde{x}$
$\tilde{x}$
Encode x
x
$\hat{y}$
Decode $\tilde{y}$
$\tilde{y}$
$\mathcal{M}(.; \boldsymbol{w}_{sup})$
Encode y
$\mathcal{R}_{out}(.; \boldsymbol{w}_{out})$
y
Parallel training


$\hat{y}$
$\tilde{x}$
$\tilde{y}$
$\mathcal{M}(.; w_{sup})$
Inference
x

Tasks combination:

$$J(\mathbb{D};\boldsymbol{w}) = \lambda_{sup}(t)\cdot J_s(\mathbb{S};\boldsymbol{w}_{sup}) + \lambda_{in}(t)\cdot J_{in}(\mathbb{F};\boldsymbol{w}_{in}) + \lambda_{out}(t)\cdot J_{out}(\mathbb{L};\boldsymbol{w}_{out})\ .$$



Tasks combination:

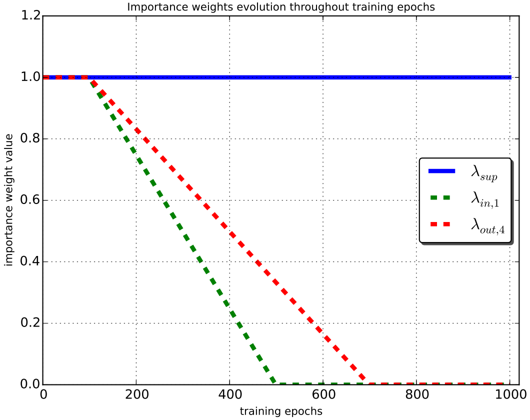
$$J(\mathbb{D}; \boldsymbol{w}) = \lambda_{sup}(t) \cdot J_s(\mathbb{S}; \boldsymbol{w}_{sup}) + \lambda_{in}(t) \cdot J_{in}(\mathbb{F}; \boldsymbol{w}_{in}) + \lambda_{out}(t) \cdot J_{out}(\mathbb{L}; \boldsymbol{w}_{out}) \ .$$

The framework training for one epoch

$\mathbb{D}$ is a *shuffled* training set. $\mathbb{B}$ a mini-batch.
for $\mathbb{B}$ in $\mathbb{D}$ **do**
    $\mathbb{B}_{\mathbb{S}} \Leftarrow$ examples of $\mathbb{B}$ that contain both $(\boldsymbol{x}, \boldsymbol{y})$.
    $\mathbb{B}_{\mathbb{F}} \Leftarrow$ all the $\boldsymbol{x}$ samples of $\mathbb{B}$.
    $\mathbb{B}_{\mathbb{L}} \Leftarrow$ all the $\boldsymbol{y}$ samples of $\mathbb{B}$.
    Make a gradient step toward $\lambda_{in} \cdot J_{in}$ using $\mathbb{B}_{\mathbb{F}}$. `# Update` $\boldsymbol{w}_{in}$
    Make a gradient step toward $\lambda_{out} \cdot J_{out}$ using $\mathbb{B}_{\mathbb{L}}$. `# Update` $\boldsymbol{w}_{out}$
    Make a gradient step toward $\lambda_{sup} \cdot J_s$ using $\mathbb{B}_{\mathbb{S}}$. `# Update` $\boldsymbol{w}_{sup}$
end for
Update λ_{sup}, λ_{in} and λ_{out}.

Tasks combination:

$$J(\mathbb{D};\boldsymbol{w}) = \lambda_{sup}(t) \cdot J_s(\mathbb{S};\boldsymbol{w}_{sup}) + \lambda_{in}(t) \cdot J_{in}(\mathbb{F};\boldsymbol{w}_{in}) + \lambda_{out}(t) \cdot J_{out}(\mathbb{L};\boldsymbol{w}_{out}) \ .$$



Linear adaptation of the importance weights during training. [Belharbi et al. 2016]

Task: Facial landmark detection. Localize 68 points (x,y).

Datasets: LFPW (1035 images), HELEN (2330 images).

Experimental results: Numerical quantification

AUC and $\mathbf{CDF_{0.1}}$ performance over LFPW test dataset with and without data augmentation. [Zhang.2014]
$\mathbf{CDF_{0.1}} \approx 95\%$ cascaded networks + multiple supervised datasets.

	No augmentation		with augmentation	
	AUC	$\mathbf{CDF_{0.1}}$	**AUC**	$\mathbf{CDF_{0.1}}$
Mean shape	68.78%	30.80%	77.81%	22.33%
MLP	76.34%	46.87%	-	-
MLP + in	77.13%	54.46%	80.78%	67.85%
MLP + out	80.93%	66.51%	81.77%	67.85%
MLP + in + out	**81.51%**	**69.64%**	**82.48%**	**71.87%**

AUC and $\mathbf{CDF_{0.1}}$ performance over HELEN test dataset with and without data augmentation. [Zhang.2014]
$\mathbf{CDF_{0.1}} \approx 95\%$ cascaded networks + multiple supervised datasets.

	No augmentation		With augmentation	
	AUC	$\mathbf{CDF_{0.1}}$	**AUC**	$\mathbf{CDF_{0.1}}$
Mean shape	64.60%	23.63%	64.76%	23.23%
MLP	76.26%	52.72%	-	-
MLP + in	77.08%	54.84%	79.25%	63.33%
MLP + out	79.63%	66.60%	80.48%	65.15%
MLP + in + out	**80.40%**	**66.66%**	**81.27%**	**71.51%**

LFPW test set. Red segments: ground truth $\longleftrightarrow$ prediction. **Top row: MLP**. **Bottom row: MLP+in+out**.

Conclusion:

- Generic regularization scheme for structured output problems based on transfer learning.
- Exploit input/output unlabeled data.
- Speedup convergence and improve generalization.
- Code at github:

https://github.com/sbelharbi/structured-output-ae

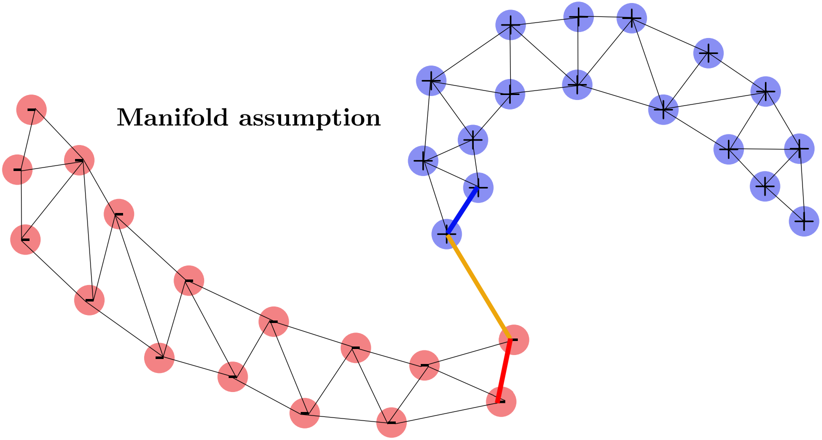
Perspectives:

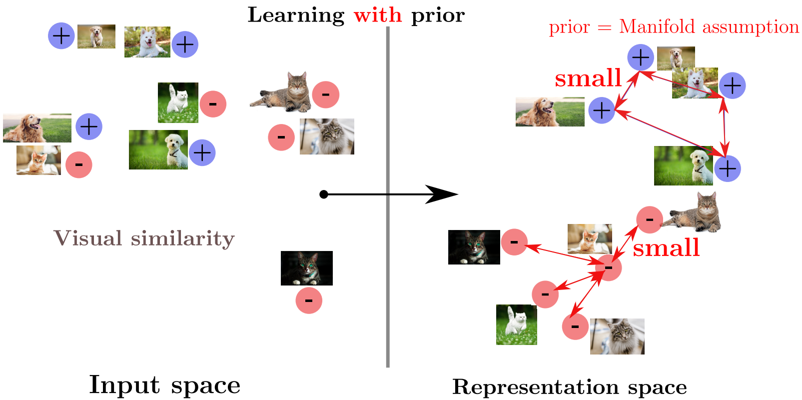
- Adapt the importance weight according to the train/validation error.
  ⇒ Toward automatic schedules.
- Use generative models to learn the output structure (VAEs, GANs).

Publications:

- S. Belharbi, R. Hérault, C. Chatelain and S. Adam. ***Deep Neural Networks Regularization for Structured Output Prediction***, Neurocomputing, vol. 281C, pp. 169-177, 2018.
- S. Belharbi, R. Hérault, C. Chatelain, S. Adam. ***Deep Multi-Task Learning with evolving weights***. European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN) (talk). 2016.
- S. Belharbi, C. Chatelain, R. Hérault, S. Adam. ***Learning Structured Output Dependencies using Deep Neural Networks***. Deep Learning workshop, International Conference on Machine Learning (ICML), 2015.

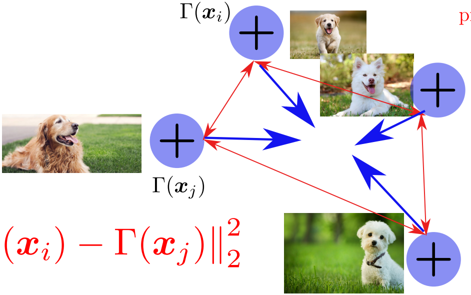
Prior knowledge for classification


Manifold assumption



- Our goal:
  Learn **invariant representations** within each class (class-wise).
- Related to:
  Linear discriminant analysis (Fisher criterion)[Sugiyama, 07], metric learning (Siamese networks).

Learning with prior



prior = Manifold assumption

$$\min \|\Gamma(\boldsymbol{x}_i) - \Gamma(\boldsymbol{x}_j)\|_2^2$$

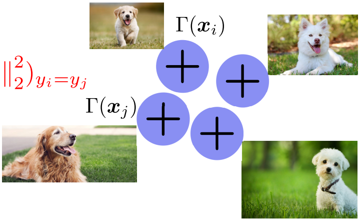
Representation space

Learning with prior

prior = Manifold assumption

$$\min \bar{\sum}(\|\Gamma(\boldsymbol{x}_i) - \Gamma(\boldsymbol{x}_j)\|_2^2)_{y_i = y_j}$$

Regularization



Representation space

Training objective:

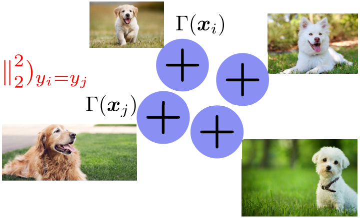
$$J(\mathbb{D}; \boldsymbol{\theta}) = \underbrace{\bar{\sum_{(\boldsymbol{x}, y) \in \mathbb{D}}} \mathcal{C}_{sup}(\mathcal{M}(\boldsymbol{x}_i), y_i)}_{\text{standard classification loss } J_{sup}} + \lambda \underbrace{\bar{\sum_{(\boldsymbol{x}, y) \in \mathbb{D}}} (\|(\Gamma(\boldsymbol{x}_i), \Gamma(\boldsymbol{x}_j)\|_2^2)_{y_i = y_j}}_{\text{invariance loss } J_r} .$$

Learning with prior

prior = Manifold assumption

$$\min \sum (\|\Gamma(\boldsymbol{x}_i) - \Gamma(\boldsymbol{x}_j)\|_2^2)_{y_i = y_j}$$

Regularization



Representation space

Training objective:

$$J(\mathbb{D}; \boldsymbol{\theta}) = \underbrace{\sum_{(\boldsymbol{x},y) \in \mathbb{D}} \mathcal{C}_{sup}(\mathcal{M}(\boldsymbol{x}_i), y_i)}_{\text{standard classification loss } J_{sup}} + \lambda \underbrace{\sum_{(\boldsymbol{x},y) \in \mathbb{D}} (\|(\Gamma(\boldsymbol{x}_i), \Gamma(\boldsymbol{x}_j)\|_2^2)_{y_i = y_j}}_{\text{invariance loss } J_r} .$$

Learning representations in a neural network for classification:

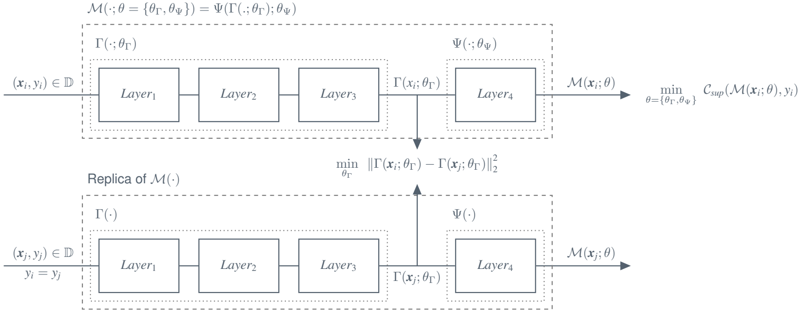
Proposed approach > Formulation

Training objective:

$$J(\mathbb{D};\boldsymbol{\theta}) = \underbrace{\sum_{(\boldsymbol{x},y)\in\mathbb{D}} \mathcal{C}_{sup}(\mathcal{M}(\boldsymbol{x}_i), y_i)}_{\text{standard classification loss } J_{sup}} + \lambda \underbrace{\sum_{(\boldsymbol{x},y)\in\mathbb{D}} (\|\Gamma(\boldsymbol{x}_i) - \Gamma(\boldsymbol{x}_j)\|_2^2)_{y_i=y_j}}_{\text{invariance loss } J_r} \, .$$

Training strategy

$\mathbb{D}$ is a training set.
$\mathbb{B}_s$ a mini-batch. $\mathbb{B}_r$ a mini-batch of all the possible pairs in $\mathbb{B}_s$.
OP_s an optimizer of the supervised term. OP_r an optimizer of the dissimilarity term.
`max_epochs`: maximum epochs. λ is a regularization weight.
for $i = 1$ to `max_epoch` **do**
    Shuffle $\mathbb{D}$. Then, split it into mini-batches.
    for $(\mathbb{B}_s, \mathbb{B}_r)$ in $\mathbb{D}$ **do**
        Make a gradient step toward J_{sup} using $\mathbb{B}_s$ and OP_s.
        Make a gradient step toward J_r using $\mathbb{B}_r$ and OP_r.
    end for
end for



Constraining the intermediate learned representations to be similar over a decomposed network $\mathcal{M}(\cdot)$ during the *training phase*.

Benchmarks: 10 classes.

Samples from training set of each benchmark. *Top row*: **mnist-std** benchmark. *Middle row*: **mnist-noise** benchmark. *Bottom row*: **mnist-img** benchmark (MNIST + CIFAR 10).

☞ Study the effect of the size of train set: $1k$, $3k$, $5k$, $50k$ and $100k$.

Models: two each one has 4 layers.

- **Multilayer perceptron (*mlp*)**: $1200 - 1200 - 200$.
- **LeNet convolutional network (*lenet*)**: $(20, 5 \times 5), (50, 5 \times 5), 500$.

Reference to layers (from input to output): h_1, h_2, h_3, h_4.

Empirical results:

- Apply the regularization at the last hidden layer ($h3$).

Results on **mnist-noise** and **mnist-img** using *lenet*:

Model/train data size	*1k*	*3k*	*5k*	*100k*
	Test error			
	mnist-std			
lenet	7.27 ± 0.033	4.02 ± 0.073	2.90 ± 0.058	-
lenet + *reg.*	$\mathbf{5.05 \pm 0.115}$	$\mathbf{2.85 \pm 0.082}$	$\mathbf{2.37 \pm 0.105}$	-
	mnist-noise			
lenet	10.72 ± 0.116	6.39 ± 0.032	5.11 ± 0.012	2.011 ± 0.018
lenet + *reg.*	$\mathbf{7.74 \pm 0.148}$	$\mathbf{4.62 \pm 0.059}$	$\mathbf{3.98 \pm 0.167}$	$\mathbf{1.64 \pm 0.116}$
	mnist-img			
lenet	15.34 ± 0.124	8.66 ± 0.024	6.46 ± 0.033	2.55 ± 0.007
lenet + *reg.*	$\mathbf{11.18 \pm 0.290}$	$\mathbf{6.61 \pm 0.212}$	$\mathbf{5.65 \pm 0.310}$	$\mathbf{2.21 \pm 0.032}$

Mean $\pm$ standard deviation error over validation and test set of the benchmarks *mnist-noise* and *mnist-img* using *lenet* model (regularization applied over the layer h_3). (**bold font indicates lowest error.**)

Conclusion:

- ☞ Our proposal helps improving the network generalization (small train set).
- ☞ Toward more explicit constraints/priors.

Publications:

- ☞ S. Belharbi, C. Chatelain, R. Hérault and S. Adam. ***Neural Networks Regularization Through Class-wise Invariant Representation Learning***, Under modification. arxiv.org/abs/1709.01867, 2018.

Transfer learning for medical domain

Medical application

Problem setup: L3 slice localization in CT scans

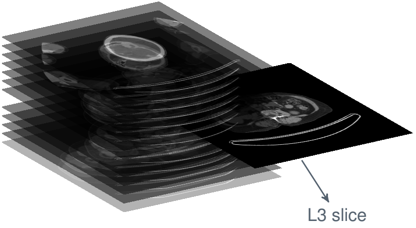
Context: Collaboration with Henri-Becquerel center at Rouen (cancer).

Main goal: Estimate the sarcopenia[1] level from a computerized tomography (CT) scan based only on the third lumbar vertebra (L3).

☞ A CT scan is stack of N slices (2D images). ☞ N is variable.

☞ In a CT scan, a specific slice is selected to represent the L3.

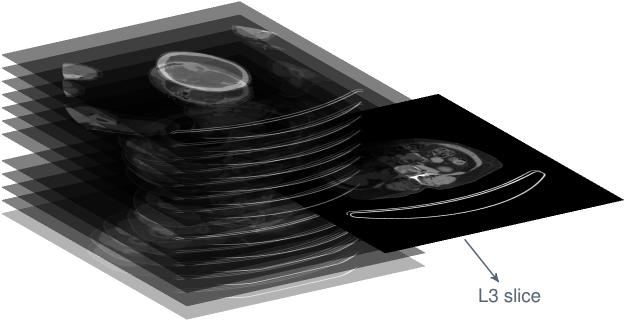
$\Rightarrow$ Need to locate the slice representing the third lumbar vertebra.



Find the L3 slice within a whole CT scan.

1. Sarcopenia: loss of skeletal muscle mass.

Problem setup: L3 slice localization in CT scans



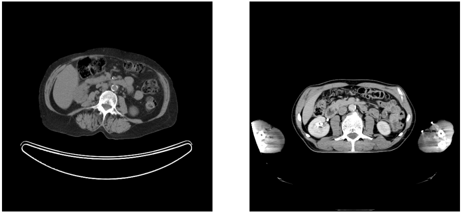
Finding the L3 slice within a whole CT scan.

- **L3CT1**:
  a dataset composed of 642 CT scans provided by Henri-Becquerel center.
- **Available annotation**:
  the position of the 3^{rd} lumber vertebra. (i.e., the number of the correct slice in the CT scan)

Problem setup: L3 slice localization in CT scans

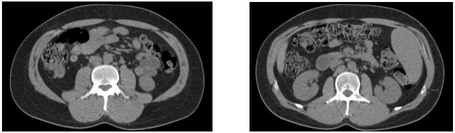
Problems:

⚠ Inter-patients variability.



L3 slices from two different patients: [Left] Patient A. [Right] Patient B.

⚠ Visual similarity of the vertebrae slices of the same patient.



Two slices from the same patient: [Left] an L3 slice. [Right] a non L3 slice.

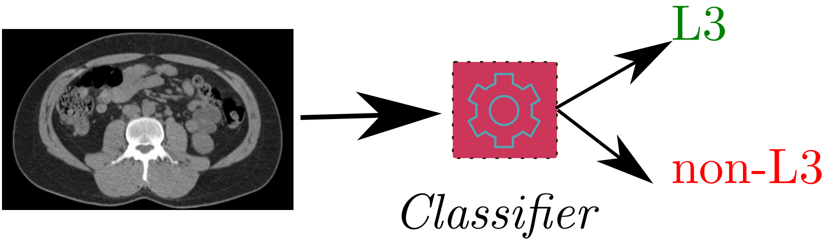
☞ The need to use the context to localize the L3 slice.

☞ **Machine Learning**!

Classification (discrete value) [X]

Classify each slice for: “L3” or “Not L3”:

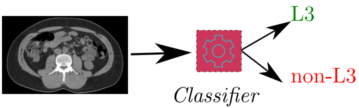
- ☞ Simple. ☺
- ⚠ No context. ☹

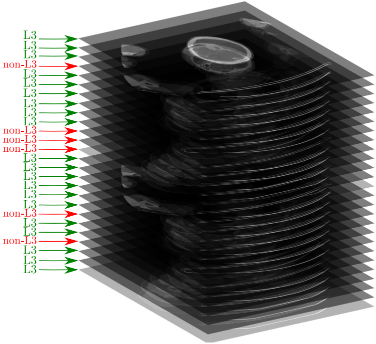


Classification (discrete value) [X]

Classify each slice for: “L3” or “Not L3”:

- Simple. ☺
- ⚠ No context. ☹



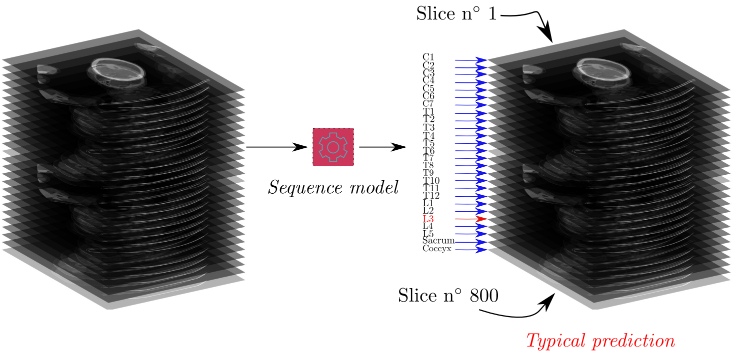


Typical prediction (no context)

Sequence labeling [X]

Label all the slices (vertebrae): L1, L2, L3, . . . :

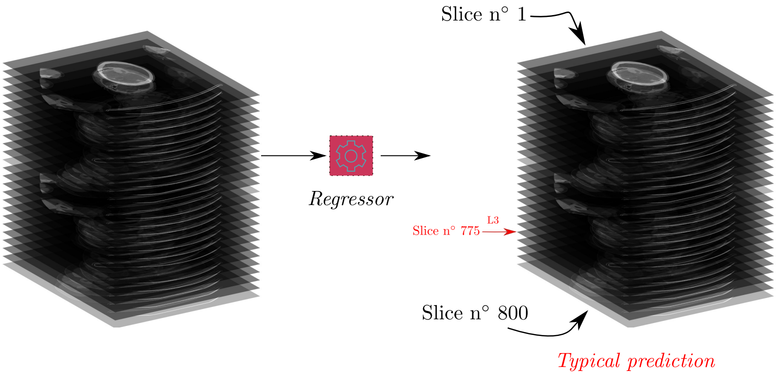
- Global analysis: context. ☺
- Existing work with promising results. ☺
- ⚠ Requires labeling more than one slice. ☹



Regression (real value) [✓]

Predict the height (position) of the L3 slice inside the CT scan:

- Global analysis: context. ☺
- Requires labeling only the L3 slice position. ☺



Which model for regression?

- State of the art in computer vision: Deep learning, convolutional neural network (CNN).
- ⚠ Requires fixed input size (when using dense layers).
- ⚠ Needs a large number of training samples.

Issues

- ⚠ High dimension input: $\underbrace{1 \text{ scan} = N \times 512 \times 512}_{\text{Problem 1: large input space}}$, with $400 < N < 1200$.
- ⚠ Implies: $\underbrace{\textit{Variability}}_{\text{Problem 2: Different input size}}$ of the height of each scan (depends on N).
- ⚠ Dataset with annotated L3 position: $\underbrace{\text{642 patients}}_{\text{Problem 3: few training data}}$. (L3CT1 dataset)

Which model for regression?

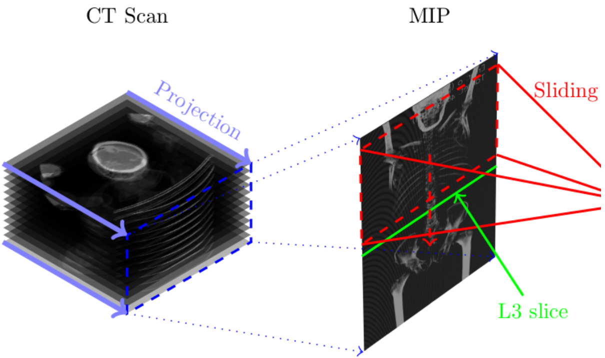
- State of the art in computer vision: Deep learning, convolutional neural network (CNN).
- ⚠ Requires fixed input size (when using dense layers).
- ⚠ Needs a large number of training samples.

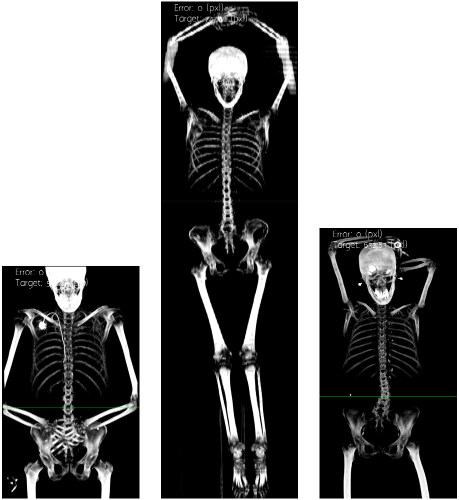
Issues

- ⚠ High dimension input: $\underbrace{1 \text{ scan} = N \times 512 \times 512}_{\text{Problem 1: large input space}}$, with $400 < N < 1200$.
- ⚠ Implies: $\underbrace{\textit{Variability}}_{\text{Problem 2: Different input size}}$ of the height of each scan (depends on N).
- ⚠ Dataset with annotated L3 position: $\underbrace{642 \text{ patients}}_{\text{Problem 3: few training data}}$. (L3CT1 dataset)

Problem 1: High dimension input

- $131M$ inputs for one example (large input dimension):
  - Frontal or lateral **Maximum Intensity Projection (MIP)**.
- $512 \times 512 \times N \Longrightarrow 512 \times N$.
- Preserves pertinent information (skeletal structure).



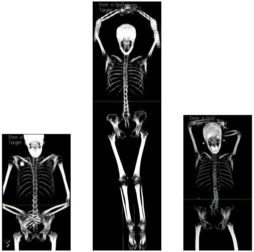


Examples of normalized frontal MIP images with the L3 slice position.

Problem 2: Different input size

Classical problem in computer vision

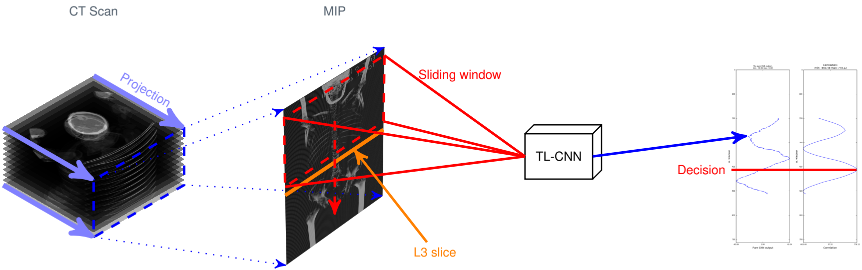
- Sliding window technique
- Post-processing



Examples of normalized frontal MIP images with the L3 slice position.

Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window



1 MIP transformation

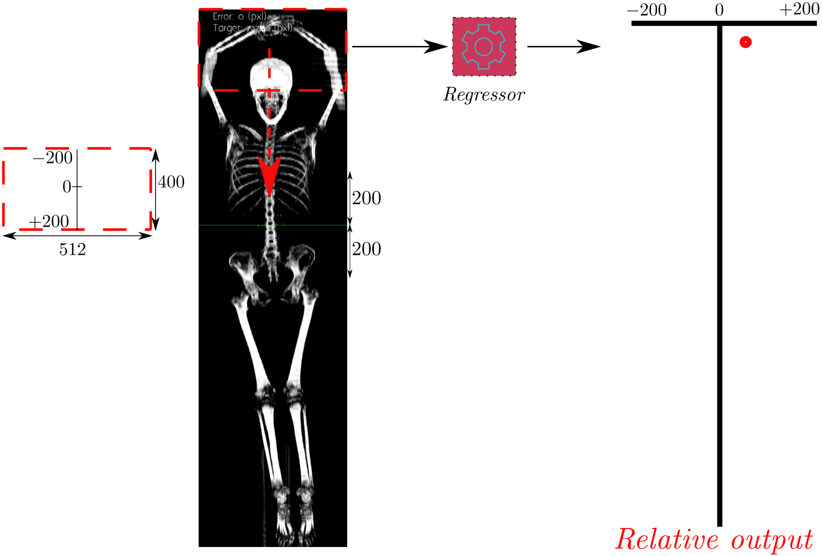
2 CNN prediction

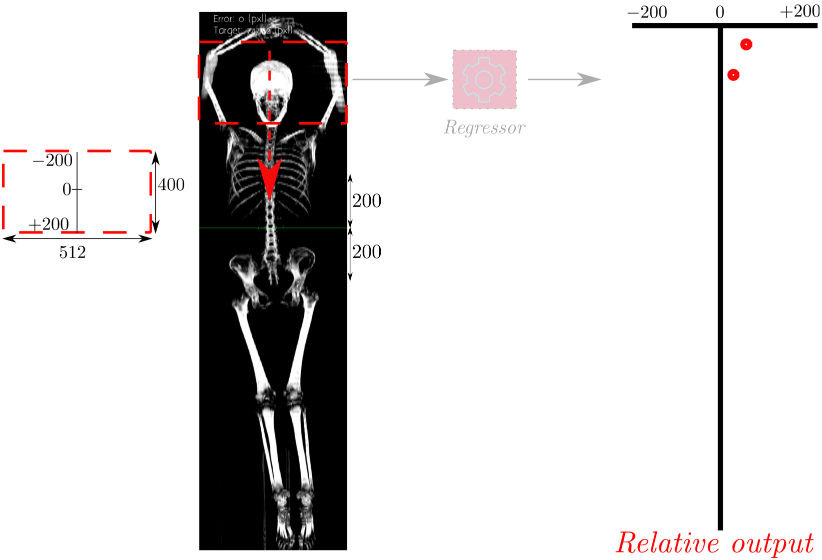
3 Post processing (Correlation)

Sliding window

Proposed approach: Regression for L3 localization

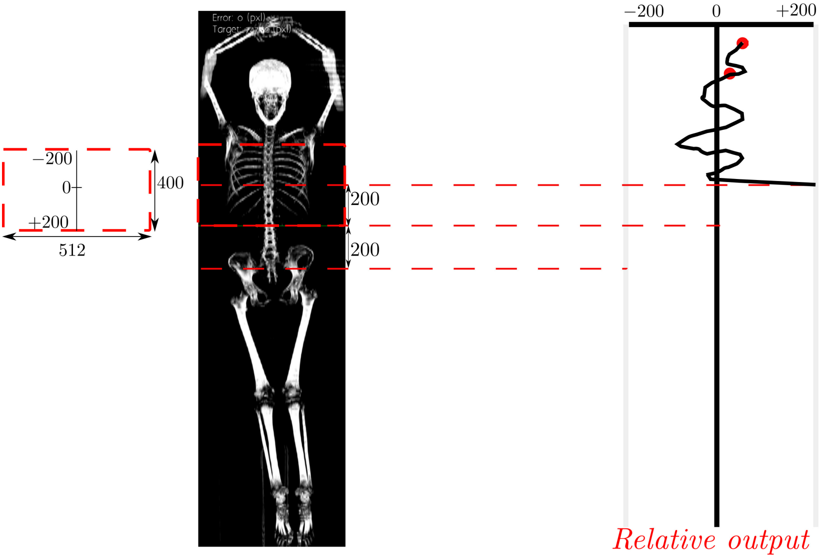
Issue 2: Different input size > Solution: Sliding window




−200
0
+200
400
512
Error: 0 (pxl)
200
200
Regressor
−200
0
+200
Relative output

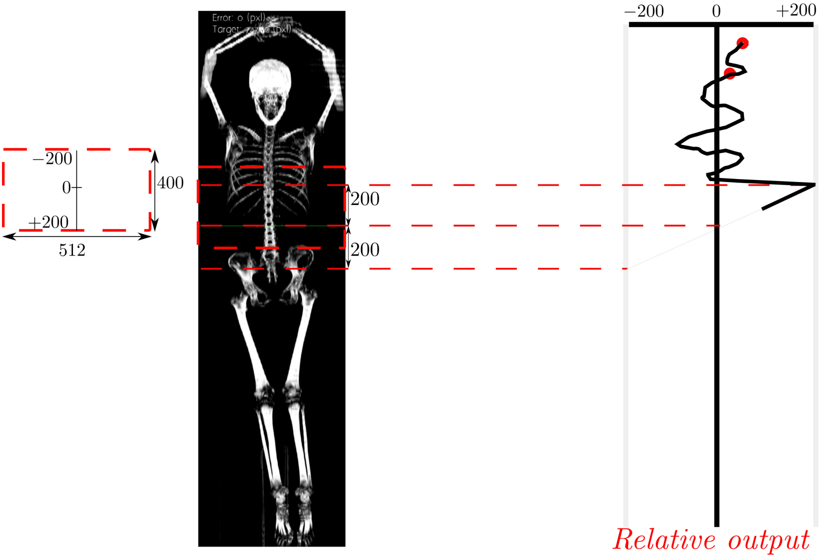
Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window



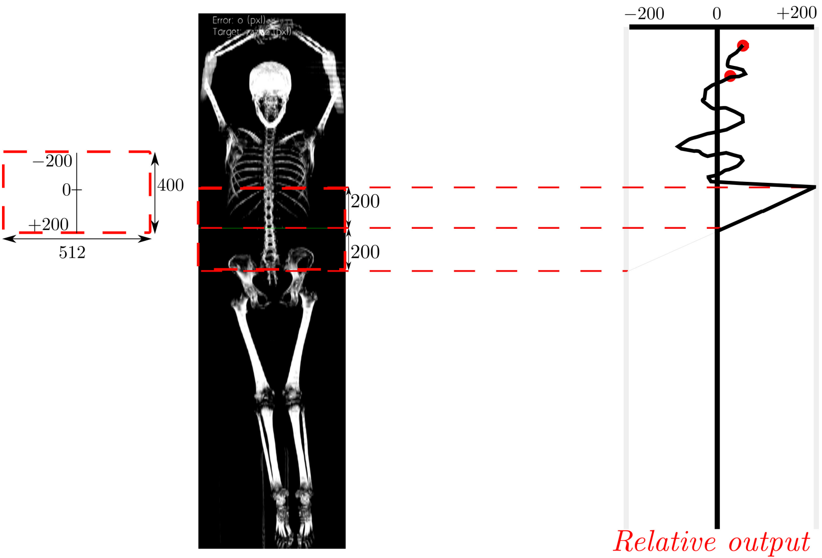
Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window



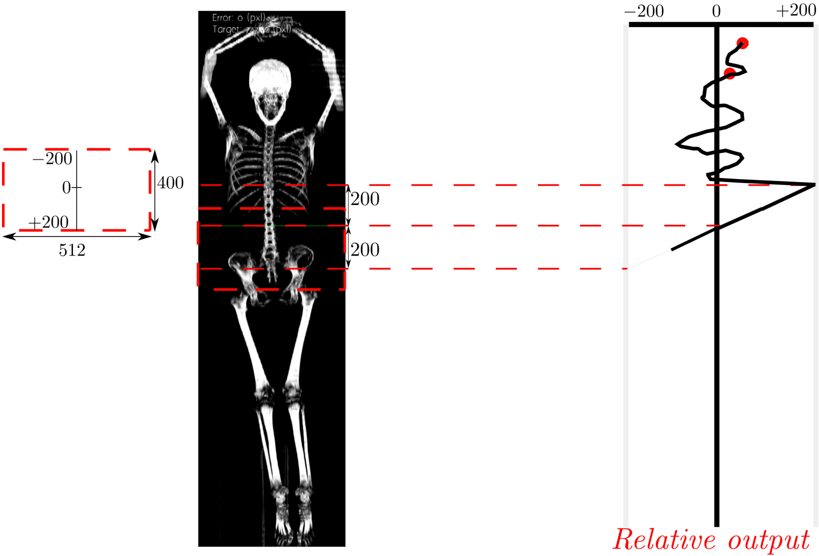
Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window



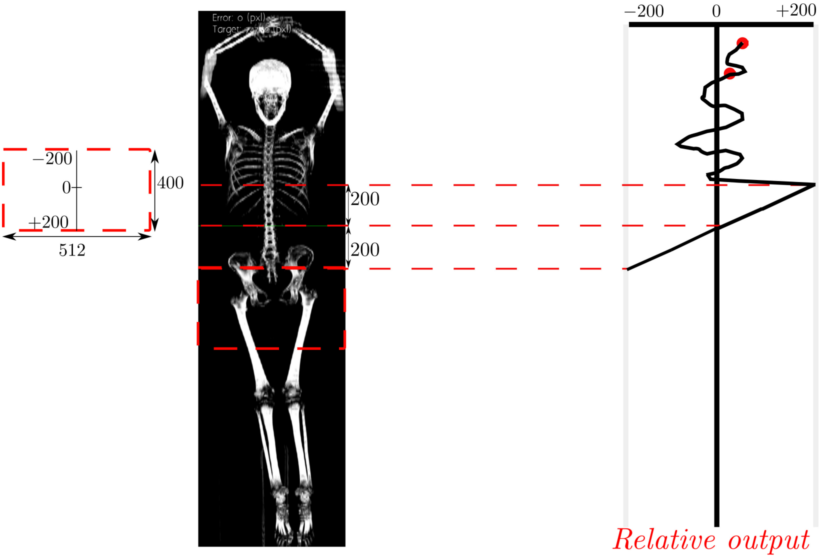
Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window



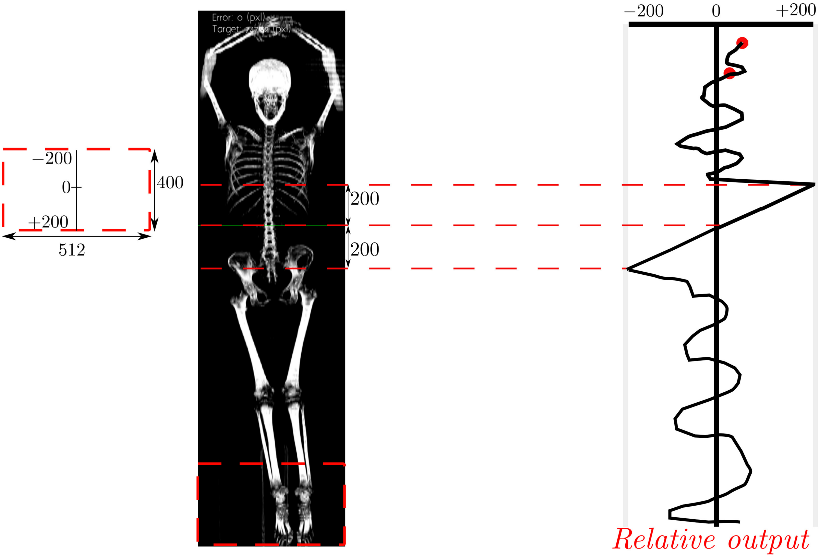
Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window



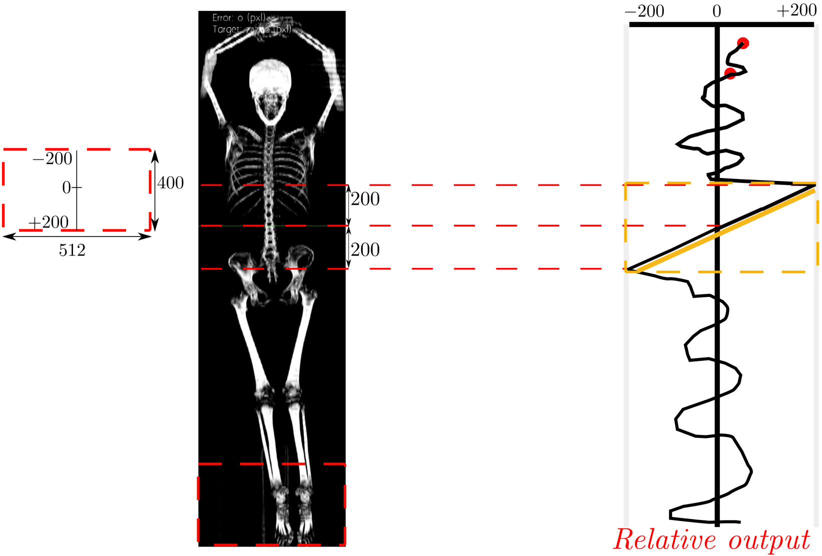
Proposed approach: Regression for L3 localization

Issue 2: Different input size > Solution: Sliding window

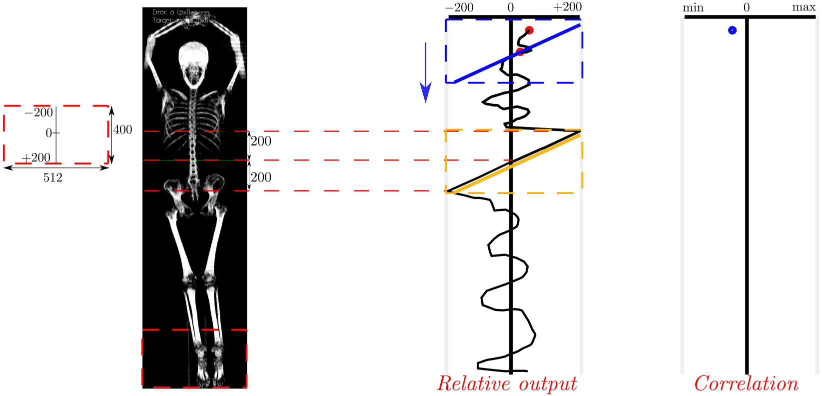


Proposed approach: Regression for L3 localization

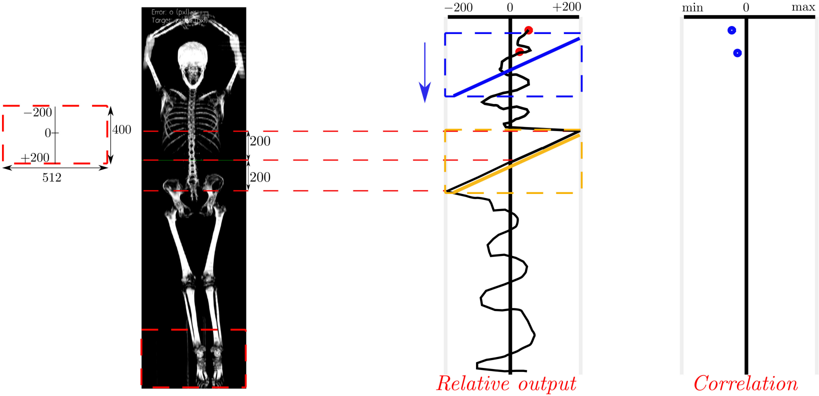
Issue 2: Different input size > Solution: Sliding window



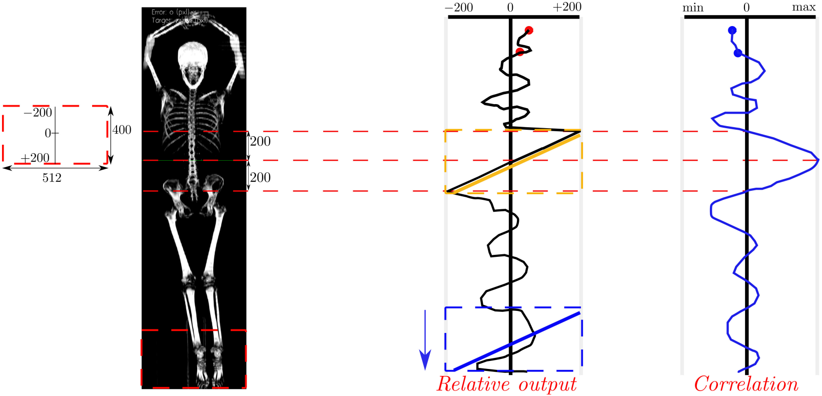
Issue 2: Different input size > Solution: Sliding window

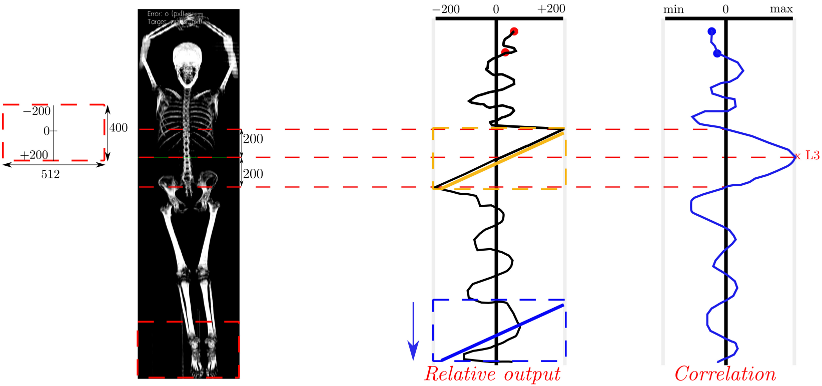


Issue 2: Different input size > Solution: Sliding window



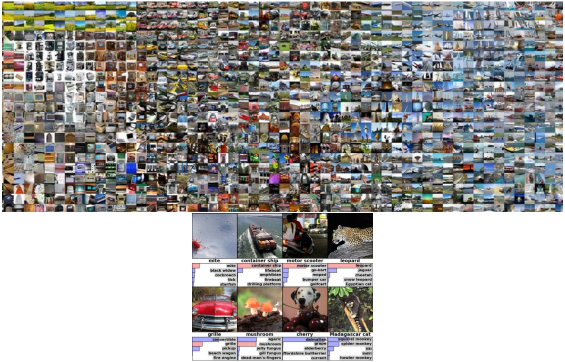
Issue 2: Different input size > Solution: Sliding window




−200
0
+200
400
512
200
200
−200
0
+200
min
0
max
x L3
Relative output
Correlation

Problem 2: Few data (642 patients)

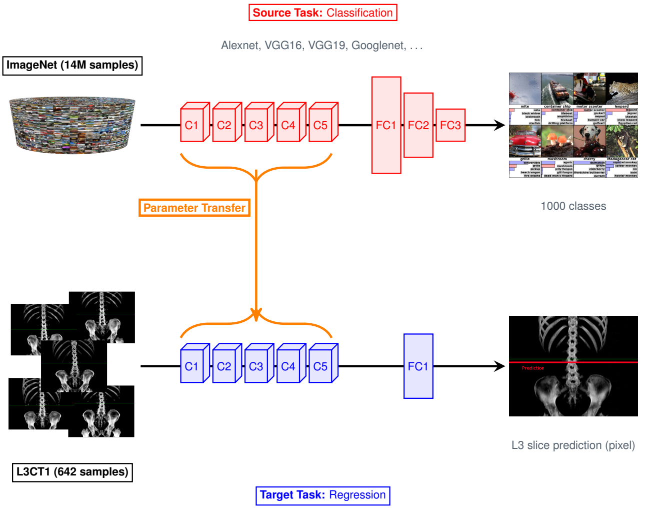
- Use pre-trained CNNs over **large datasets**
  - Alexnet, GoogleNet, VGG16, VGG19, . . . for **classification**
  - Pre-trained models over ImageNet: 14 millions of natural images [Fei-Fei and Russakovsky 2013].



Source task with abundant data.

Proposed approach: Regression for L3 localization

Issue 3: Lack of data > Solution: Transfer learning



System training using transfer learning.

Cross-validation:

	RF500	CNN4	Pre-trained			
			Alexnet	VGG16	VGG19	Googlenet
Average cross-validation error (5 folds) (slice)	10.50 ± 10.80	2.78 ± 2.48	2.45 ± 2.42	$\mathbf{1.82 \pm 2.32}$	1.83 ± 1.83	2.54 ± 4.22
Number of parameters	–	55 K	2 M	14 M	20 M	6[1] M
Average processing time (second/CT scan) (K40)	–	04.46	06.37	13.28	16.02	17.75[1]

RF500 (random forest with 500 decision trees), CNN4 (Homemade model), and Alexnet/VGG16/VGG19/GoogleNet (Pre-trained models).

Possible speedup: reduce the number of sampled windows $\Rightarrow$ Increase stride.

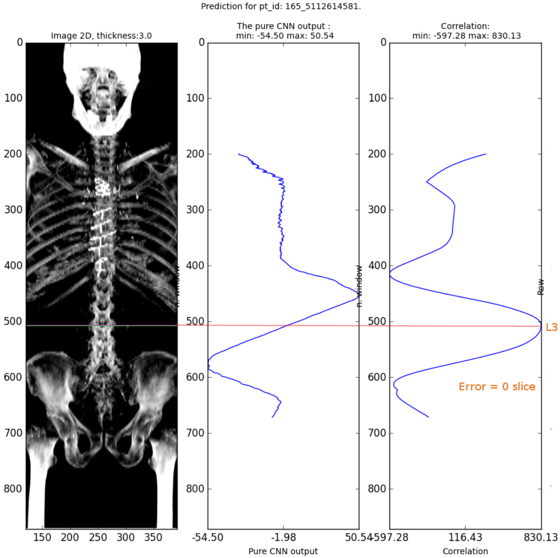
Example VGG16:

- **stride=1**: $\sim$ **13 *seconds/CT scan*** with a an error of $\mathbf{1.82 \pm 2.32}$.
- **stride=4**: $\sim$ **02 *seconds/CT scan*** with a an error of $\mathbf{1.91 \pm 2.69}$.

1. Due to implementation.

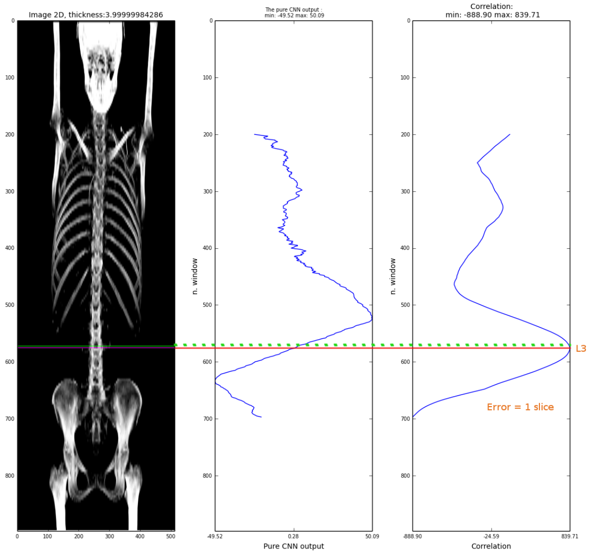
Proposed approach: Regression for L3 localization

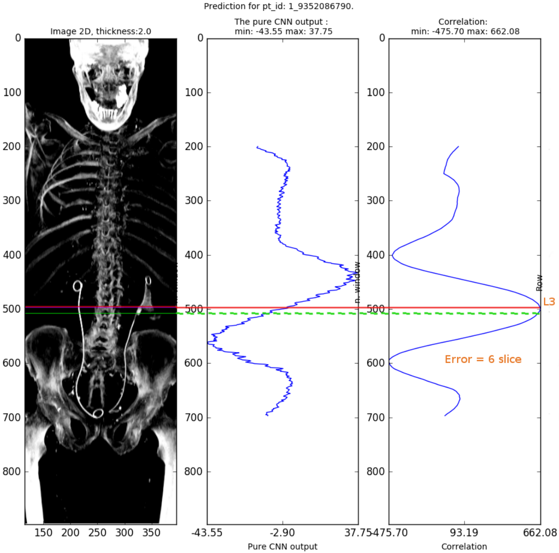
Experiments: Qualitative results



Proposed approach: Regression for L3 localization

Experiments: Qualitative results




Prediction for pt_id: 1_9352086790.
Image 2D, thickness:2.0
The pure CNN output :
min: -43.55 max: 37.75
Correlation:
min: -475.70 max: 662.08
L3
Error = 6 slice
Pure CNN output
Correlation
150 200 250 300 350
-43.55 -2.90 37.75
-475.70 93.19 662.08
0 100 200 300 400 500 600 700 800

Setup: Intra-annotator variability

- New evaluation set: 43 CT scans annotated by the same reference radiologist (who annotated the L3CT1 dataset).
- Ask 3 other radiologists to localize the L3 slice.
- Perform this experiment twice: t_1, t_2.

Errors (slices) / operator	Ragiologist $\#1$	Radiologist $\#2$	Radiologist $\#3$
t_1	0.81 ± 0.97	0.72 ± 1.51	0.51 ± 0.62
t_2	0.77 ± 0.68	0.95 ± 1.61	0.86 ± 1.30

Intra-annotator variability.

Setup: Intra-annotator variability

- New evaluation set: 43 CT scans annotated by the same reference radiologist (who annotated the L3CT1 dataset).
- Ask 3 other radiologists to localize the L3 slice.
- Perform this experiment twice: t_1, t_2.

Errors (slices) / operator	Ragiologist $\#1$	Radiologist $\#2$	Radiologist $\#3$	CNN4	VGG16
t_1	0.81 ± 0.97	0.72 ± 1.51	0.51 ± 0.62	2.37 ± 2.30	1.70 ± 1.65
t_2	0.77 ± 0.68	0.95 ± 1.61	0.86 ± 1.30	2.53 ± 2.27	1.58 ± 1.83

Performance radiologists vs. automatic systems.

- Adapted pipeline for L3 localization: pre-processing, CNN, post-processing.
- Obtained average error: 1.82 slice ($< 5mm$) (maximum error: 9 slices).
  - Average thickness of a vertebra $\approx 2.5cm \Rightarrow$ Still within the L3 vertebra.
- Learn context: sliding window (double checked using correlation: context over multiple windows.)
- Generic framework: can be easily adapted for detecting other subjects given the required annotation.
- Use of transfer learning alleviates the lack of training data.

Perspectives: ⚠ Running time of VGG16 over CPUs is time consuming.
- Possible solution: Prune unnecessary convolution filters.

Valorization:
- Integrate this work with the software of the projet "**BodyComp.AI**" (diffused to European centers for cancer treatment).
- "**BodyComp.AI**" has won one of the 2017 French Innovative Unicancer Prize.

Publications:

- S. Belharbia, C. Chatelain, R. Hérault, S. Adam, S. Thureau, M. Chastan, and R. Modzelewski. ***Spotting L3 slice in CT scans using deep convolutional network and transfer learning***, Computers in Biology and Medicine, vol. 87, pp. 95-103, 2017.

General conclusion & perspectives

- Possible improvements in the generalization of neural networks through the use of regularization based on representation learning paradigm in different applications (few training data):
  - Structured output problems: Unsupervised learning.
  - Classification: Invariant representations prior.
  - Object localization: Transfer learning.

- Improve neural networks generalization through:
  - Integrating priors/common sense.
    - Reduce the dependency to statistics.
      - Require less training data.
  - Use well studied data representations methods as hidden layers.
    - Mimic dictionary learning.

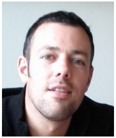
Dictionary learning:

$$\arg\min_{\boldsymbol{D}\in\mathbb{C}, \boldsymbol{r}_i\in\mathbb{R}^d} \sum_{i=1}^{N} \|\boldsymbol{x}_i - \boldsymbol{D}\boldsymbol{r}_i\|_2^2 \;, \text{ where } \mathbb{C} \equiv \{\boldsymbol{D} \in \mathbb{R}^{d\times K} : \|\boldsymbol{d}_i\|_2 \leq 1 \;\forall i = 1, \cdots, K\} \;.$$

Thank you for your attention!

Questions?

soufiane.belharbi@insa-rouen.fr
sbelharbi.github.io



Clément CHATELAIN

Romain HÉRAULT

Sébastien ADAM

In memory of
Frank ROSENBLATT
1928-1971

Computation resource

CRIANN: Myria

UFR Sciences et Techniques's data center

INSA Rouen Normandie