Probabilistic and deep learning for regression in computer vision

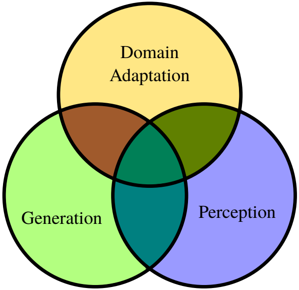
Stéphane Lathuilière & Xavier Alameda-Pineda

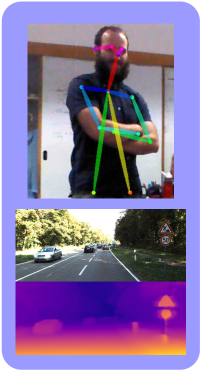
Multimedia and Human Understanding Group (MHUG),
University of Trento, Italy
Multimedia Team,
Telecom Paris, France

Preception Team,
Multidisciplinary Institute of Artificial Intelligence,
University Grenoble-Alpes,
Inria Grenoble Rhône-Alpes, France

September 9, 2019

Presenters: Stéphane





2014		2018		2019	
	PhD		Post-Doc		Assistant Professor
	Perception		**MHUG**		**Multimedia**
	Dr. R. Horaud		Prof. Nicu Sebe		Prof. Marco Cagnazzo

Presenters: Xavi

Audio-Visual Perception
Low-level Behavioral Robotics
Multi-modal Machine Learning

Job openings:
- PhD scholarships
- Post-docs
- Engineers

co-Chairman: Audio-Visual Machine Perception and Interaction for Companion Robots @ MIAI-Grenoble

Inria — PhD — 2010 — **Perception** — Dr. R. Horaud

UNIVERSITÀ DI TRENTO — Post-Doc — 2014 — **MHUG** — Prof. Nicu Sebe

2016 — Inria — Research Scientist

Tutorial Outline

Today's outline

1. Introduction [15 min] – Xavi
2. Pratrical Deep Regression Guidelines [10 min] – Xavi
3. Robust deep Regression with Probabilistic Models [20 min] – Xavi
4. Basics of Image and Video Generation [10 min] – Stéphane
5. Pose-based Human Image Generation [15 min] – Stéphane
6. Video generation: Image Animation [15 min] – Stéphane
7. Conclusions [5 min] – Stéphane

Please do not hesitate to ask questions on the fly!!!

Slides will be available at http://xavirema.eu.

Introduction

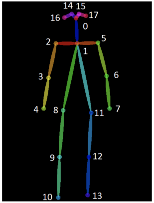
The task of regression is that of fitting a functional **relationship between two continuous variables**.

$$y = f(x; \theta), \tag{1}$$

where:

- $x \in \mathbb{R}^I$ and $y \in \mathbb{R}^O$ are input and output variables,
- f is the function used to model the regression problem,
- θ are the parameters of the function to be learned.

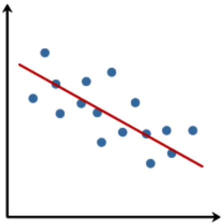
Examples:



Linear regression (simplest type):

$$\boldsymbol{y} = \mathbf{A}\boldsymbol{x} + \boldsymbol{b},$$

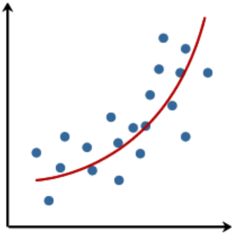
where $\mathbf{A} \in \mathbb{R}^{O \times I}$, $\boldsymbol{b} \in \mathbb{R}^{O}$, and therefore $\boldsymbol{\theta} = \{\mathbf{A}, \boldsymbol{b}\}$.



Non-linear regression – more generic (it could be a deep network):

$$\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta}),$$

$f : \mathbb{R}^{I} \to \mathbb{R}^{O}$.



Need to **fit the parameters** $\rightarrow$ Set a loss and an optimisation problem.

*Images from Laerd Statistics

Fitting the parameters

The parameters $\boldsymbol{\theta}$ should be estimated, we require:

- a set $\mathcal{T} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N}$ of input-output training pairs,
- a sample loss $\mathcal{L} : \mathbb{R}^O \times \mathbb{R}^O \to \mathbb{R}^+$.

Fitting the parameters

The parameters $\boldsymbol{\theta}$ should be estimated, we require:

- a set $\mathcal{T} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N}$ of input-output training pairs,
- a sample loss $\mathcal{L} : \mathbb{R}^O \times \mathbb{R}^O \to \mathbb{R}^+$.

The aim is to minimise the empirical risk:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}), \qquad R(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}(f(\boldsymbol{x}_n; \boldsymbol{\theta}), \boldsymbol{y}_n). \tag{2}$$

The parameters $\boldsymbol{\theta}$ should be estimated, we require:

- a set $\mathcal{T} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N}$ of input-output training pairs,
- a sample loss $\mathcal{L} : \mathbb{R}^O \times \mathbb{R}^O \to \mathbb{R}^+$.

The aim is to minimise the empirical risk:

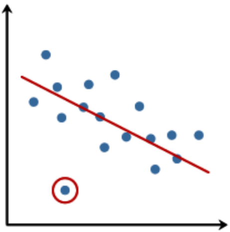
$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} R(\boldsymbol{\theta}), \qquad R(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(f(\boldsymbol{x}_n; \boldsymbol{\theta}), \boldsymbol{y}_n). \tag{2}$$

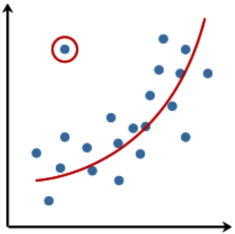
Optimisation techniques (depend on $\mathcal{L}$, f and $\boldsymbol{\theta}$):

- close-form,
- gradient-descent, stochastic GD, mini-batch GD,
- expectation-minimsation algorithm,

or a combination of the above.

An outlier does not follow the data distribution: $(\boldsymbol{x}_o, \boldsymbol{y}_o) \not\sim p(\boldsymbol{x}, \boldsymbol{y})$





Robustness to outliers:

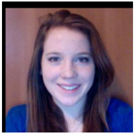
- use loss functions that reduce the impact on the training,
- detect and remove outliers in advance.

Examples of outliers

In age estimation (bad annotations):

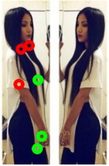

(a) 14


(b) 14


(c) 60


(d) 62

In fashion landmark detection (bad input format):



Practical Guidelines for Deep Regression

Recall: $\mathcal{T} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N}$, where $\boldsymbol{x}_n \in \mathbb{R}^I$ and $\boldsymbol{y}_n \in \mathbb{R}^O$.

In deep regression, $f(\cdot; \boldsymbol{\theta}) : \mathbb{R}^I \to \mathbb{R}^O$ is a deep neural network.

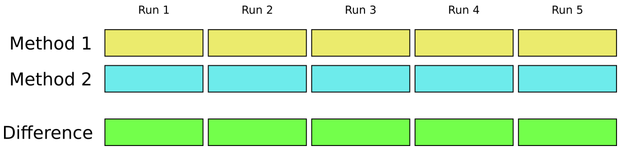
The standard loss used is the L_2 norm, leading to:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{n=1}^{N} \|f(\boldsymbol{x}_n; \boldsymbol{\theta}) - \boldsymbol{y}_n\|^2. \tag{3}$$

How do we evaluate the *significancy* of the results? How do we asses that one deep regression method is *better* than another one?

Standard evaluation "protocol": single run, report the mean (perhaps variance)
$\rightarrow$ not enough!!!

Our protocol: five runs, Wilcoxon signed-rank test, 95% confidence interval for the median of the MAE.



$$e_{1,n} = \|f_1(\boldsymbol{x}_n) - \boldsymbol{y}_n\|^2$$
$$e_{2,n} = \|f_2(\boldsymbol{x}_n) - \boldsymbol{y}_n\|^2$$

$$d_{12,n} = e_{1,n} - e_{2,n}$$

If the two methods are equivalent, the random variable $d_{12,n}$ is symmetric.

Otherwise the methods are not equivalent
$\rightarrow$ we pick the one with lower median error.

What is being tested?

For two network baselines (VGG16, ResNet51) and three tasks (head pose, facial & body landmark).

- Optimisation:
  - Four stochastic optimisation techniques.
  - Four batch sizes.
- Architecture options:
  - Fine-tuning depth.
  - Use of batch-normalisation.
  - Input and output representation of the regression (pooling, heatmap, flatten).
  - Regressed layer.
  - Use of dropout.
  - The regression loss.
- Several data pre-processing methods (depending on the data set).

This implies 600+ runs and takes roughly 64 days on TITAN X.

Lots of results


A Comprehensive Analysis of Deep Regression

1/3 of the paper are tables and figures → good luck ;)

Well-known:

- The larger BS the better.
- BN is crucial.
- Always mirror your data.

Well-known:

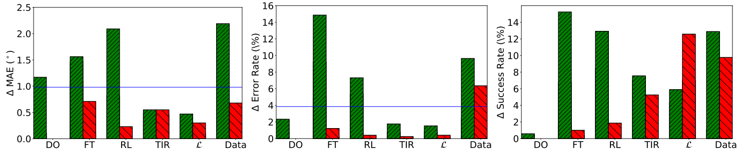
- The larger BS the better.
- BN is crucial.
- Always mirror your data.

More interesting:

- Adam is the best optimisation choice.
- How you use dropout for VGG is not that crucial, but you must use it!!!
- There is a data-dependent balance between over/underfitting when chosing the fine-tuning depth.
- Do not regress from feature maps, rather from FC/GP features (except for heat-map regression).

DeepNets are awesome because we do not need to hand-craft features (yey!).

VGG and **ResNet** relative difference of best and worst strategies.
Performance increase of recent publications.



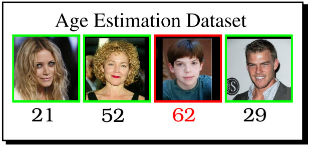
(h) Biwi **(i)** LFW+NET **(j)** MPII

DO-dropout, FT-fine tuning, RL-regressed layer, TIR-target/input repr., $\mathcal{L}$-loss, Data-Preproc.

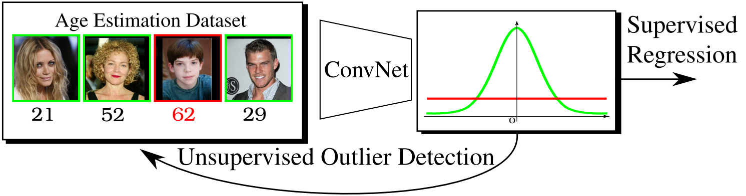
We observe a HUGE performance variation depending on the data pre-processing $\longrightarrow$ mind your pre-processing.

Robust Deep Regression with Probabilistic Models

Motivation: Clean annotations for large-scale datasets are expensive (and not realistic). How to train a deep regression method robustly to noise?

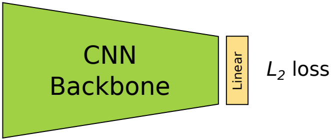


Motivation: Clean annotations for large-scale datasets are expensive (and not realistic). How to train a deep regression method robustly to noise?



Limitations of standard deep regression

Standard way: deep model + linear regression layer + L_2 loss:



(a) Standard Deep Regression Model

(b) Residual gradient

The larger the gradient, the more attention the network pays to it.

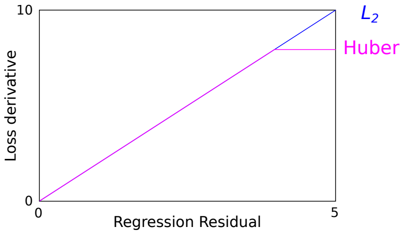
Gradient of the L_2 loss is 2δ, twice the residual.

Outliers have huge residual $\Rightarrow$ The network pays a lot of attention.

Let's take a look to existing solutions:

Existing solutions

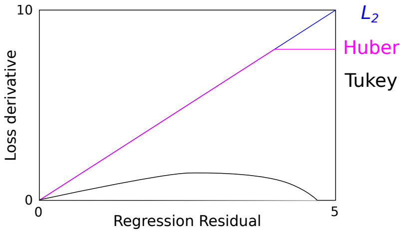
Let's take a look to existing solutions:



L_2/Huber large gradient for large δ.

Existing solutions

Let's take a look to existing solutions:

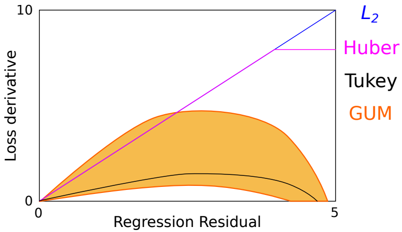


L_2/Huber large gradient for large δ.

Tukey not flexible/interpretable.

Existing solutions

Let's take a look to existing solutions:

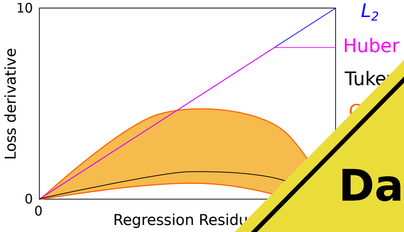


L_2/Huber large gradient for large δ.

Tukey not flexible/interpretable.

Gaussian-Uniform Mixtures (GUM) offer a **family** of interpretable losses.

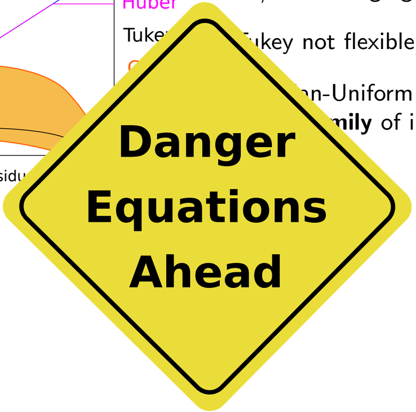
Let's take a look to existing solutions:



L_2/Huber large gradient for large δ.

ukey not flexible/interpretable.

n-Uniform Mixtures (GUM)
mily of interpretable losses.



Hypothesis: inliers $\leftrightarrow$ Gaussian outliers $\leftrightarrow$ Uniform.

$$p(y_n, x_n; \nu, \theta) = \underbrace{\rho}_{\text{Inlier prior}} \mathcal{N}(y_n - \phi(x_n;\theta); 0, \Sigma) + \underbrace{(1-\rho)}_{\text{Outlier prior}} \mathcal{U}(y_n - \phi(x_n;\theta); \gamma),$$

Gaussian Uniform Mixtures

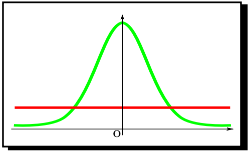
 21  52  62  29

Hypothesis: inliers $\leftrightarrow$ Gaussian outliers $\leftrightarrow$ Uniform.

$$p(y_n, x_n; \nu, \theta) = \underbrace{\rho}_{\text{Inlier prior}} \mathcal{N}(y_n - \phi(x_n; \theta); 0, \Sigma) + \underbrace{(1-\rho)}_{\text{Outlier prior}} \mathcal{U}(y_n - \phi(x_n; \theta); \gamma),$$

$\phi(\cdot; \theta)$: forward with weights θ
$\nu = \{\rho, \Sigma, \gamma\}$: parameters of GUM

Challenge: How to learn θ and ν?

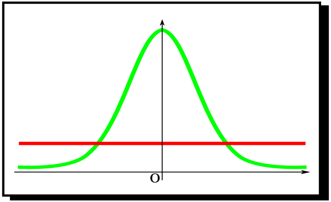


Main idea: Expectation-maximisation (EM).

Latent variable $z_n = 1$ inlier, $z_n = 0$ outlier.

E-step: $r_n(\nu^{(r)}) = p(z_n = 1 | x_n, y_n, \nu^{(r)})$.

$$r_n(\nu^{(r)}) = \frac{\rho^{(r)} \mathcal{N}(y_n - \phi(x_n; \theta^{(r)}); 0, \Sigma^{(r)})}{\rho^{(r)} \mathcal{N}(y_n - \phi(x_n; \theta^{(r)}); 0, \Sigma^{(r)}) + (1 - \rho^{(r)}) \mathcal{U}(y_n - \phi(x_n; \theta); \gamma^{(r)})}$$



How to train? (II)

Main idea: Expectation-maximisation (EM).

M-ν step: update ν:

$$\rho^{(r+1)} = \frac{1}{N} \sum_{n=1}^{N} r_n(\nu^{(r)})$$

$$\Sigma^{(r+1)} = \frac{1}{N} \sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n; \theta^{(r)}))(y_n - \phi(x_n; \theta^{(r)}))^\top$$

γ is updated to fit the variance of the detected outliers.

M-θ step: update θ by minimising

$$\mathcal{L}_{\mathsf{GUM}} = \sum_{n=1}^{N} r_n(\nu^{(r)}) \|y_n - \phi(x_n; \theta)\|^2.$$

How to train? (II)

Main idea: Expectation-maximisation (EM).

M-ν step: update ν:

$$\rho^{(r+1)} = \frac{1}{N} \sum_{n=1}^{N} r_n(\nu^{(r)})$$

$$\Sigma^{(r+1)} = \frac{1}{N} \sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n; \theta^{(r)}))(y_n - \phi(x_n; \theta^{(r)}))^\top$$

γ is updated to fit the variance of the detected outliers.

M-θ step: update θ by minimising

$$\mathcal{L}_{\mathsf{GUM}} = \sum_{n=1}^{N} r_n(\nu^{(r)}) \|y_n - \phi(x_n; \theta)\|^2.$$

What is wrong?

Expected complete-data log-likelihood

M-step starts from:

$$
\begin{aligned}
Q(\nu, \nu^{(r)}) &= \sum_{n=1}^{N} \mathbb{E}_{p(z_n|x_n, y_n, \nu^{(r)})}\{p(z_n, x_n, y_n|\nu)\} \\
&= \mathcal{C} + \left(\rho - \frac{\log|\Sigma|}{2}\right) \sum_{n=1}^{N} r_n(\nu^{(r)}) + (1-\rho)\sum_{n}(1 - r_n(\nu^{(r)})) \\
&- \frac{1}{2}\sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top \Sigma^{-1}(y_n - \phi(x_n;\theta)) \\
&+ \sum_{n=1}^{N}(1 - r_n(\nu^{(r)})) \log \mathcal{U}(y_n - \phi(x_n;\theta);\gamma)
\end{aligned}
$$

Expected complete-data log-likelihood

M-step starts from:

$$
\begin{aligned}
Q(\nu, \nu^{(r)}) &= \sum_{n=1}^{N} \mathbb{E}_{p(z_n|x_n,y_n,\nu^{(r)})}\{p(z_n, x_n, y_n|\nu)\} \\
&= \mathcal{C} + \left(\rho - \frac{\log|\Sigma|}{2}\right) \sum_{n=1}^{N} r_n(\nu^{(r)}) + (1-\rho)\sum_{n}(1 - r_n(\nu^{(r)})) \\
&- \frac{1}{2}\sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top \Sigma^{-1}(y_n - \phi(x_n;\theta)) \\
&+ \sum_{n=1}^{N}(1 - r_n(\nu^{(r)})) \log \mathcal{U}(y_n - \phi(x_n;\theta); \gamma)
\end{aligned}
$$

Then,

$$
\frac{\partial Q}{\partial \theta} = 0 \Longrightarrow \mathcal{L}_{\mathsf{GUM}} = \sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top \Sigma^{-1}(y_n - \phi(x_n;\theta))
$$

and not:

$$
\mathcal{L}_{\mathsf{GUM}} = \sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top (y_n - \phi(x_n;\theta)) = \sum_{n=1}^{N} r_n(\nu^{(r)})\|y_n - \phi(x_n;\theta)\|^2.
$$

Expected complete-data log-likelihood

M-step starts from:

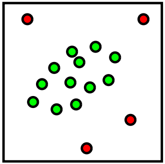
$$
\begin{aligned}
Q(\nu, \nu^{(r)}) &= \sum_{n=1}^{N} \mathbb{E}_{p(z_n|x_n, y_n, \nu^{(r)})}\{p(z_n, x_n, y_n|\nu)\} \\
&= \mathcal{C} + \left(\rho - \frac{\log|\Sigma|}{2}\right) \sum_{n=1}^{N} r_n(\nu^{(r)}) + (1-\rho)\sum_n (1 - r_n(\nu^{(r)})) \\
&- \frac{1}{2}\sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top \Sigma^{-1}(y_n - \phi(x_n;\theta)) \\
&+ \sum_{n=1}^{N}(1 - r_n(\nu^{(r)}))\log \mathcal{U}(y_n - \phi(x_n;\theta);\gamma)
\end{aligned}
$$

Then,

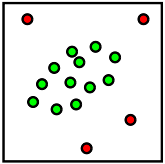
$$
\frac{\partial Q}{\partial \theta} = 0 \Longrightarrow \mathcal{L}_{\mathsf{GUM}} = \sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top \Sigma^{-1}(y_n - \phi(x_n;\theta))
$$

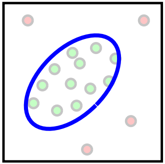
and not:

$$
\mathcal{L}_{\mathsf{GUM}} = \sum_{n=1}^{N} r_n(\nu^{(r)})(y_n - \phi(x_n;\theta))^\top (y_n - \phi(x_n;\theta)) = \sum_{n=1}^{N} r_n(\nu^{(r)})\|y_n - \phi(x_n;\theta)\|^2.
$$



1. Set of regression errors
$\{y_n - \phi(x_n;\theta)\}_{n=1}^{N}$
Inliers & Outliers

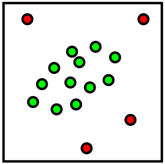


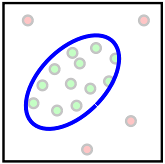


1. Set of regression errors $\{y_n - \phi(x_n;\theta)\}_{n=1}^{N}$
Inliers & Outliers

2. Compute Σ

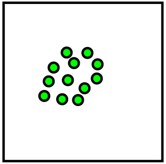
Why Σ^{-1} matters?





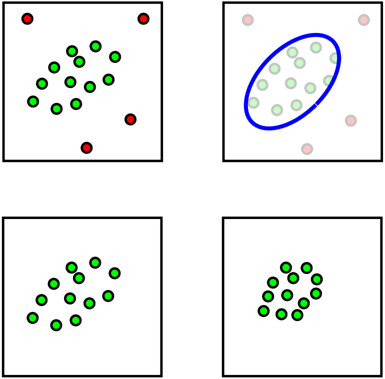
1. Set of regression errors $\{y_n - \phi(x_n;\theta)\}_{n=1}^{N}$
Inliers & Outliers

2. Compute Σ



Right: Normalise error by Σ^{-1}.

Why Σ^{-1} matters?



1. Set of regression errors $\{y_n - \phi(x_n;\theta)\}_{n=1}^{N}$
Inliers & Outliers

2. Compute Σ

Right: Normalise error by Σ^{-1}.

Left: Do not normalise error.

If we normalise, error directions that occur often are ignored.

Results with synthetic outliers (I)

Original data: LFW and Net[1] facial landmark datasets.

Three types of synthetic outliers:

NGO Normally Generated Outliers: a % of the landmarks are contaminated with a Gaussian displacement.

l-UGO Local Uniformly Generated Outliers: same as NGO with uniform displacement.

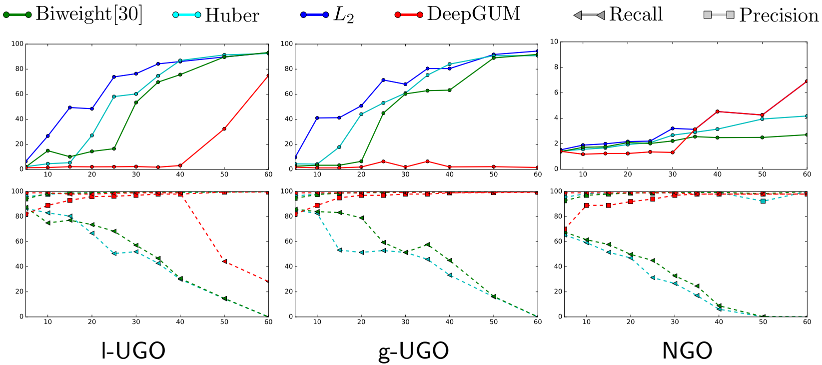
g-UGO Global UGO: all landmarks of a % of images is contaminated with uniform displacement.

The % goes from 0 to 60. We report the failure rate (the method's error is larger than 5% of the image size).

Because we contaminate the data, we know which observations are outliers, and we can compute the precision and recall.

[1] Sun, Y., Wang, X., Tang, X.: Deep convolutional network cascade for facial point detection. In: CVPR (2013)

Results with synthetic outliers (II)



- DeepGUM is much better (for uniform) than the other robust losses.
- For Gaussian outliers is comparable/better/worse dependeing.
- The breakdown point is quite high.

Results on Fashion Landmark Detection (I)

Fashion landmark dataset.[2]

Mean absolute error on the upper-body subset of FLD, average per landmark. Legend: left (L) and right (R) collar (C), sleeve (S) and hem (H).
*** Denotes statistical significance with $p < 0.001$.

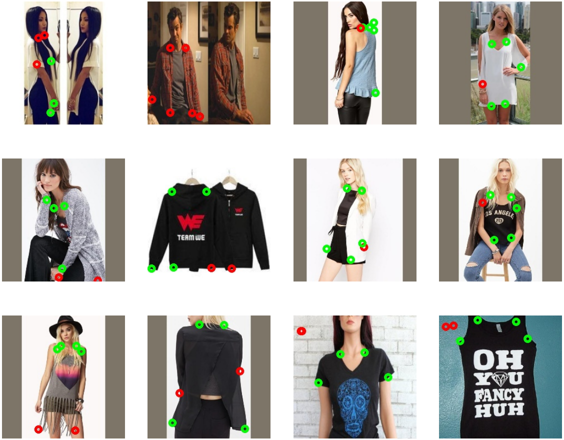
Method	Upper-body landmarks						
	LC	RC	LS	RS	LH	RH	Avg.
DFA[3] (L_2)	15.90	15.90	30.02	29.12	23.07	22.85	22.85
DFA (5 VGG)	10.75	10.75	20.38	19.93	15.90	16.12	15.23
L_2	12.08	12.08	18.87	18.91	16.47	16.40	15.80
Huber[4]	14.32	13.71	20.85	19.57	20.06	19.99	18.08
Biweight[5]	13.32	13.29	21.88	21.84	18.49	18.44	17.88
DeepGUM	11.97***	11.99***	18.59***	18.50***	16.44***	16.29***	15.63***

[2] Liu, Z., Luo, P., Qiu, S., Wang, X., Tang, X.: Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In: CVPR (2016)

[3] Liu, Z., Yan, S., Luo, P., Wang, X., Tang, X.: Fashion Landmark Detection in the Wild. In: ECCV (2016)

[4] Huber, P.J.: Robust estimation of a location parameter. The annals of mathematical statistics pp. 73101 (1964)

[5] Belagiannis, V., Rupprecht, C., Carneiro, G., Navab, N.: Robust optimization for deep regression. In: ICCV (2015)



Outliers detected by DeepGUM

Results on Head Pose Estimation

Experiments on the McGill dataset.[6]

Report the Mean Absolute Error and the Root Mean Squared Error.
*** Denotes statistical significance with $p < 0.001$.
† Denotes the use of extra training data.

Method	MAE	RMSE
Xiong et al.[7†]	-	29.81 ± 7.73
Zhu and Ramanan[8†]	-	35.70 ± 7.48
Demirkus et al.[†]	-	12.41 ± 1.60
Drouard et al.[9]	12.22 ± 6.42	23.00 ± 9.42
L_2	8.60 ± 1.18	12.03 ± 1.66
Huber	8.11 ± 1.08	11.79 ± 1.59
Biweight	7.81 ± 1.31	11.56 ± 1.95
DeepGUM***	7.61 ± 1.00	11.37 ± 1.34

[6] Demirkus, M., Precup, D., Clark, J.J., Arbel, T.: Hierarchical temporal graphical model for head pose estimation and subsequent attribute classification in real-world videos. CVIU (2015)

[7] Xiong, X., De la Torre, F.: Supervised descent method and its applications to face alignment. In: CVPR. (2013)

[8] Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. In: CVPR. pp. 28792886 (2012)

[9] Drouard, V., Horaud, R., Deleforge, A., Ba, S., Evangelidis, G.: Robust head-pose estimation based on partially-latent mixture of linear regressions. TIP 26, 14281440 (2017)

Experiments on the CACD dataset.[10]

Report the Mean Absolute Error.
*** Denotes statistical significance with $p < 0.001$.

Method	MAE
L_2	5.75
Huber	5.59
Biweight	5.55
Dex[11]	5.25
DexGUM***	5.14
DeepGUM***	5.08

[10] Chen, B.C., Chen, C.S., Hsu, W.H.: Cross-age reference coding for age-invariant face recognition and retrieval. In: ECCV (2014)
[11] Rothe, R., Timofte, R., Van Gool, L.: Deep expectation of real and apparent age from a single image without facial landmarks. IJCV (2016)

(o) 14

(p) 14

(q) 14

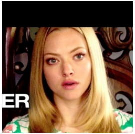
(r) 16

(s) 20

(t) 23

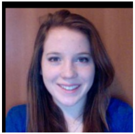
(u) 49

(v) 51

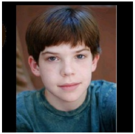


(w) 60

(x) 60



(y) 60



(z) 62

Outliers detected by DeepGUM

From low- to high-dimensional regression

In classical regression, even when addressed with deep architectures, we have input dimension much higher than output (age estimation, head pose estimation, landmarks).

Recent CNN allow to address regression problems with both input and output being high-dimensional spaces (image-to-image). The second half of the talk deals with that.