Deep learning for classification is handled using standard losses and output representations, but this is not (yet) the case when it comes to regression.
Fredrik Gustafsson (PhD student, Uppsala University)  
Martin Danelljan (post-doc, ETH)


The problem we are interested in – regression using DNNs

**Supervised regression:** learn to predict a continuous output (target) value $\mathbf{y}^* \in \mathcal{Y} = \mathbb{R}^K$ from a corresponding input $\mathbf{x}^* \in \mathcal{X}$, given a training set $\mathcal{D}$ of i.i.d. input-output data

$$\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N, \quad (x_n, y_n) \sim p(x, y).$$

**Deep neural network (DNN):** a function $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$, parameterized by $\theta \in \mathbb{R}^P$, that maps an input $\mathbf{x} \in \mathcal{X}$ to an output $f_\theta(\mathbf{x}) \in \mathcal{Y}$.

Generally applicable, but we have (so far) mainly worked with examples from computer vision and robotics.

**Input space** $\mathcal{X}$: Space of images or point clouds.

**Output space** $\mathcal{Y} = \mathbb{R}^K$: $\mathcal{Y} = \mathbb{R}^2$ for image-coordinate regression, $\mathcal{Y} = \mathbb{R}_+$ for age estimation, $\mathcal{Y} = \mathbb{R}^4$ for 2D bounding-box regression.
Intuitive preview of our construction

A general regression method with a clear probabilistic interpretation.

Let us first note that with a probabilistic take on regression, the task is to learn the conditional target density $p(y \mid x)$.

We create and train an energy-based model (EBM) of the conditional target density $p(y \mid x)$, allowing for highly flexible target densities to be learned directly from data.

1D toy illustration showing that we can learn multi-modal and asymmetric distributions, i.e. our model is flexible.
Aim: Create an awareness of how we can use deep neural networks for regression and show that energy-based models are useful in this context.

1. Intuitive preview
2. Regression using deep neural networks
3. Energy-based models
4. Our construction
5. Training
6. Experiments
Four existing approaches

1. Direct regression
2. Probabilistic regression
3. Confidence-based regression
4. Regression-by-classification
1. Direct regression (I/II)

Train a DNN $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ to directly predict the target $y^* = f_\theta(x^*)$.

Learn the parameters $\theta$ by minimizing a loss function $\ell(f_\theta(x_i), y_i)$, penalizing discrepancy between prediction $f_\theta(x_i)$ and ground truth $y_i$

$$\hat{\theta} = \arg\min_\theta J(\theta),$$

where

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(x_i), y_i).$$

Common choices for $\ell$ are the $L^2$ loss, $\ell(\hat{y}, y) = \|\hat{y} - y\|_2^2$, and the $L^1$ loss, $\ell(\hat{y}, y) = \|\hat{y} - y\|_1$. 
1. Direct regression (II/II)

Minimizing

\[ J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(x_i), y_i) \]

then corresponds to minimizing the negative log-likelihood
\[ \sum_{i=1}^{N} - \log p(y_i | x_i; \theta), \text{ for a specific model } p(y | x; \theta) \text{ of the conditional target density.} \]

Ex: The \( L^2 \) loss corresponds to a fixed-variance Gaussian model:
\[ p(y | x; \theta) = \mathcal{N}(y; f_\theta(x), \sigma^2). \]
Why not explicitly employ this probabilistic perspective and try to create more flexible models $p(y \mid x; \theta)$ of the conditional target density $p(y \mid x)$?

One idea is to restrict the parametric model to unimodal distributions such as Gaussian or Laplace.

**Probabilistic regression:** train a DNN $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ to predict the parameters $\phi$ of a certain family of probability distributions $p(y; \phi)$, then model $p(y \mid x)$ with

$$p(y \mid x; \theta) = p(y; \phi(x)), \quad \phi(x) = f_\theta(x).$$

The parameters $\theta$ are learned by minimizing $\sum_{i=1}^{N} - \log p(y_i \mid x_i; \theta)$. 


Ex: A general 1D Gaussian model can be realized as:

\[ p(y \mid x; \theta) = \mathcal{N}(y; \mu_\theta(x), \sigma^2_\theta(x)), \]

where the DNN is trained to output

\[ f_\theta(x) = \left( \mu_\theta(x), \log \sigma^2_\theta(x) \right)^T \in \mathbb{R}^2. \]

The negative log-likelihood \( \sum_{i=1}^{N} - \log p(y_i \mid x_i; \theta) \) then corresponds to

\[ J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \frac{(y_i - \mu_\theta(x_i))^2}{\sigma^2_\theta(x_i)} + \log \sigma^2_\theta(x_i). \]
The quest for improved regression accuracy has also led to the development of more specialized methods.

Confidence-based regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ to predict a scalar confidence value $f_\theta(x, y)$, and maximize this quantity over $y$ to predict the target

$$y^* = \arg\max_y f_\theta(x^*, y)$$

Key to this approach is that $f_\theta(x, y)$ depends on both the input $x$ and the target $y$.

The parameters $\theta$ are learned by generating pseudo ground truth confidence values $c(x_i, y_i, y)$, and minimizing a loss function $\ell(f_\theta(x_i, y), c(x_i, y_i, y))$. 
Discretize the output space $\mathcal{Y}$ into a finite set of $C$ classes and use standard classification techniques...
High-level description of our idea

Confidence-based regression give impressive results, but:

1. it require important (and tricky) task-dependent design choices (e.g. how to generate the pseudo ground truth labels)
2. and usually lack a clear probabilistic interpretation.

Probabilistic regression is straightforward and generally applicable, but:

1. it can usually not compete in terms of regression accuracy.

Our construction combines the benefits of these two approaches while removing the problems above.
Background – Energy-based models (EBM)

An energy-based models (EBM) specifies a probability density

\[
p(x; \theta) = \frac{e^{f_\theta(x)}}{Z(\theta)}, \quad Z(\theta) = \int e^{f_\theta(x)} d\mathbf{x},
\]

explicitly parameterized by the scalar function \( f_\theta(x) \).

By defining \( f_\theta(x) \) using a deep neural network, \( p(x; \theta) \) becomes expressive enough to learn practically any density from observed data.


The EBM allows for the full predictive power of the DNN to be exploited, enabling us to learn

- multimodal and
- asymmetric densities
directly from data.

The cost of the flexibility is that the normalization constant (partition function)

$$Z(\theta) = \int e^{f_\theta(x)} dx$$

is intractable, which complicates

- evaluating $p(y \mid x; \theta)$ and
- sampling from $p(y \mid x; \theta)$. 
A general regression method with a **clear probabilistic interpretation** in the sense that we learn a model $p(y \mid x, \theta)$ **without** requiring $p(y \mid x, \theta)$ to belong to a particular family of distributions.

Let the DNN be a function $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that maps an input-output pair $\{x_i, y_i\}$ to a scalar value $f_\theta(x_i, y_i) \in \mathbb{R}$.

Define the resulting (flexible) probabilistic model as a conditional EBM

$$
p(y \mid x, \theta) = \frac{e^{f_\theta(x, y)}}{Z(x, \theta)}, \quad Z(x, \theta) = \int e^{f_\theta(x, \tilde{y})} d\tilde{y}
$$
The DNN $f_{\theta}(x, y)$ that specifies the conditional EBM can be trained using methods for fitting a density $p(y | x; \theta)$ to observed data \[ \{(x_n, y_n)\}_{n=1}^{N}. \]

The most straightforward method is to minimize the negative log-likelihood

$$
\mathcal{L}({\theta}) = - \sum_{i=1}^{N} \log p(y_i | x_i; {\theta})
$$

$$
= \sum_{i=1}^{N} \log \left( \int e^{f_{\theta}(x_i, \tilde{y})} d\tilde{y} \right) - f_{\theta}(x_i, y_i) - f_{\theta}(x_i, y_i).
$$

**Challenge:** Requires the normalization constant to be evaluated (the integral is intractable)...
Solution 1 – Importance sampling

\[ p(y | x, \theta) = \frac{e^{f_\theta(x, y)}}{Z(x, \theta)}, \quad Z(x, \theta) = \int e^{f_\theta(x, \tilde{y})} d\tilde{y} \]

The parameters \( \theta \) are learned by minimizing \( \sum_{n=1}^{N} -\log p(y_n | x_n; \theta) \).

Use importance sampling to evaluate \( Z(x, \theta) \):

\[
-\log p(y_i | x_i; \theta) = \log \left( \int e^{f_\theta(x_i, y)} dy \right) - f_\theta(x_i, y_i)
\]

\[
= \log \left( \int \frac{e^{f_\theta(x_i, y)}}{q(y)} q(y) dy \right) - f_\theta(x_i, y_i)
\]

\[
\approx \log \left( \frac{1}{M} \sum_{k=1}^{M} \frac{e^{f_\theta(x_i, y^{(k)})}}{q(y^{(k)})} \right) - f_\theta(x_i, y_i), \quad y^{(k)} \sim q(y).
\]

Use a Gaussian mixture (centered around the measurements) as proposal.
**Noise Contrastive Estimation (NCE)** is a parameter estimation method for loglinear models, which avoids calculation of the partition function (normalization constant) or its derivatives at each training step.


This is precisely what we need!

NCE entails learning to discriminate between observed data examples and samples drawn from a noise distribution.
Using NCE for regression

Using NCE for regression entails training the DNN $f_\theta(x, y)$ by minimizing

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^{N} J_i(\theta),$$

$$J_i(\theta) = \log \frac{\exp\left\{ f_\theta(x_i, y_i^{(0)}) - \log q(y_i^{(0)} | y_i) \right\}}{\sum_{m=0}^{M} \exp\left\{ f_\theta(x_i, y_i^{(m)}) - \log q(y_i^{(m)} | y_i) \right\}},$$

where $y_i^{(0)} \triangleq y_i$, and $\{y_i^{(m)}\}_{m=1}^{M}$ are $M$ samples drawn from a noise distribution $q(y | y_i)$ that depends on the true target $y_i$.

**Interpretation:** $J(\theta)$ is the softmax cross-entropy loss for a classification problem with $M + 1$ classes.

A simple choice for $q(y | y_i)$ is a mixture of $K$ Gaussians centered at $y_i$,

$$q(y | y_i) = \frac{1}{K} \sum_{k=1}^{K} \mathcal{N}(y; y_i, \sigma_k^2 I).$$
The EBM is trained by having to discriminate between the given label $y_i$ (red box) and noise samples $\{y^{i,m}\}_{m=1}^M$ (blue boxes).
Allowing NCE to account for noise in the annotations

We have slightly generalized NCE to explicitly account for noise in the annotation process.

Given a label $y_i$ (red box), the EBM is trained by having to discriminate between $y_i + \nu_i$ (yellow box) and noise samples $\{y^{i,m}\}_{m=1}^M$ (blue boxes).
Allowing NCE to account for noise in the annotations

The DNN $f_\theta(x, y)$ is still trained by minimizing

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^{N} J_i(\theta), \quad J_i(\theta) = \log \frac{\exp \{ f_\theta(x_i, y_i^{(0)}) - \log q(y_i^{(0)} | y_i) \}}{\sum_{m=0}^{M} \exp \{ f_\theta(x_i, y_i^{(m)}) - \log q(y_i^{(m)} | y_i) \}},$$

but $y_i^{(0)}$ is now defined as

$$y_i^{(0)} \triangleq y_i + \nu_i.$$

The true target $y_i$ is thus perturbed with $\nu_i \sim q_\beta(y)$, where

$$q_\beta(y) = \frac{1}{K} \sum_{k=1}^{K} \mathcal{N}(y; 0, \beta \sigma_k^2 I).$$

This is how we can account for possible inaccuracies in the annotation process producing $y_i$. 

---

Train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ to predict $f_\theta(x, y)$ and model $p(y \mid x)$ with

$$p(y \mid x, \theta) = \frac{e^{f_\theta(x, y)}}{Z(x, \theta)}, \quad Z(x, \theta) = \int e^{f_\theta(x, \tilde{y})} d\tilde{y}.$$

The parameters $\theta$ are learned by minimizing $\sum_{i=1}^{N} - \log p(y_i \mid x_i; \theta)$.

Given a test input $x^\star$, we predict the target $y^\star$ by maximizing $p(y \mid x^\star; \theta)$

$$y^\star = \arg \max_y p(y \mid x^\star; \theta) = \arg \max_y f_\theta(x^\star, y).$$

By designing the DNN $f_\theta$ to be differentiable w.r.t. targets $y$, the gradient $\nabla_y f_\theta(x^\star, y)$ can be efficiently evaluated using auto-differentiation.

Use gradient ascent to find a local maximum of $f_\theta(x^\star, y)$, starting from an initial estimate $\hat{y}$. 

23/26
Experiments – Visual tracking


**Task (visual tracking):** Estimate a bounding box of a target object in every frame of a video. The target object is defined by a given box in the first video frame.

Show Movie!

**Task:** Detect objects from sensor data (here laser), estimate their size and position in the 3D world.

Key perception task for self-driving vehicles and autonomous robots.

The combination of **probabilistic models** and **deep neural networks** is very exciting and promising.

Aim: Create an awareness of how we can use deep neural networks for regression and show that energy-based models are useful in this context.

- Introduced an EBM for regression using DNNs
- Solved the training problem using
  - Importance sampling
  - Generalized noise contrastive estimation
- State-of-the-art performance on challenging regression problems using images and laser point clouds.