Deep transformation models for tackling complex probabilistic regression problems

Goal: Predict a flexible outcome distribution


Beate Sick *  
EBPI, University of Zurich &  
IDP, Zurich University of Applied Sciences  
Email: beate.sick@uzh.ch, sick@zhaw.ch

Torsten Hothorn  
EBPI, University of Zurich  
Email: torsten.hothorn@uzh.ch

Oliver Dürr *  
IOS, Konstanz University of Applied Sciences  
Email: oliver.duerr@htwg-konstanz.de
Non-probabilistic versus probabilistic regression DL models

Deterministic

\( \hat{y} = 117 \)

Probabilistic

\[ f(y) = 117 \]

\[ \theta(x) = \arg\min_{\mu} \sum_{i=1}^{N} \log(N(y_i; \mu(x_i), \sigma)) \]

\[ N(\mu(x_i), \sigma) \]

systolic blood pressure (sbp)

x (age)

systolic blood pressure (sbp)

age
Have a look on more complex conditional probability distributions

Image credits: Lucas Kook
How to model complex distributions?

- Use a mixture model (e.g. mixture Gaussians)
- Use a transformation model!
Idea of a transformation model

Idea: we fit a transformation function $h$ that transforms the flexible conditional outcome $f_i^Y(y)$ to an easy (here $N(0,1)$) distribution $f^Z(z)$

$$NLL = \sum_i -\log(f_i^Y(y_i)) = \sum_i -\log \left( f^z(z_i) \cdot \left| \frac{\partial h_{\theta_i}}{\partial y} \right|_{y_i} \right)$$

"change of variable" formula

$f^Z(z)$

$z_i = h_{\theta_i}(y_i)$

$f_i^Y(y)$
For non-Gaussian CPDs we need a non-linear transformation.
Using Bernstein polynomials to approximate the transformation $h$

$$z_x = h_{\theta(x)}(\tilde{y}) = \sum_{k=1}^{M} \frac{\vartheta_k(x)}{M+1} Be_k(\tilde{y})$$

$\tilde{y} \in [0,1]$  

Bernstein polynomials have nice properties:

- They can approximate every function on the support $[0; 1]$
- The order $M$ controls the flexibility
- Its bijective, i.e. monotone increasing, if parameters $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_M$

Most Likely Transformation (MLT) 2017 by T. Hothorn, L. Möst, P. Bühlmann  
Our deep transformation model

\[ f_i^Y(y) \]

\[ f_1: \; \bar{y} = a(x) \cdot y - b(x) \]

\[ f_2: \; \bar{z} = \frac{1}{M + 1} \sum_{i=0}^{M} Be_i(\sigma(\bar{y})) \theta_i(x) \]

\[ f_3: \; z = a(x) \cdot \bar{z} - \beta(x) \]

\[ h_\theta(x) = f_3, \alpha, \beta \circ f_2, \theta_0, \ldots, \theta_M \circ \sigma \circ f_1, \alpha, \beta \]

\[ \text{NLL} = \sum_i -\log \left( f^z(z_i) \cdot \left| \frac{\partial h_\theta(x_i)}{\partial y} \right|_{\bar{y}_i} \right) \]

R implementation: [https://github.com/tensorchiefs/dl_mlt](https://github.com/tensorchiefs/dl_mlt)
Architecture of our Deep transformation model

\[
h_\theta(x) = f_{3,\alpha_x,\beta_x} \circ f_{2,\vartheta_0_x,\ldots,\vartheta_M x} \circ \sigma \circ f_{1,\alpha_x,\beta_x} \quad \text{NLL} = \sum_i -\log \left( f^z_i(z_i) \cdot \left| \frac{\partial h_{\theta(x_i)}}{\partial y} \right|_{y_i} \right)
\]

Ensuring a monotone increasing \( h \):

Positive slope:
\( a > 0 \)

Increasing Bernstein coefficients:
\[
\begin{align*}
\vartheta_0 &= \gamma_0 \\
\vartheta_1 &= \vartheta_0 + e^{\gamma_1} \\
& \quad \ldots \\
\vartheta_M &= \vartheta_{M-1} + e^{\gamma_M} \\
\rightarrow & \ \vartheta_0 < \vartheta_1 < \ldots < \vartheta_M
\end{align*}
\]

Positive slope:
\( \alpha > 0 \)
Application: Predict CPD for age based on an image

For 10 randomly picked images from test set with true ages (1 year, 30 years, 90 years) the predicted CPD is shown. The solid lines correspond to shown images.
Application: Benchmarking our model

**TABLE I**

Comparison of prediction performance (test NLL, smaller is better) on regression benchmark UCI datasets. The best method for each dataset is bolded, as are those with standard errors that overlap with the standard errors of the best method.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>N</th>
<th>DL_MLT</th>
<th>NGBoost</th>
<th>MC Dropout</th>
<th>Deep Ensembles</th>
<th>Gaussian Process</th>
<th>MDN</th>
<th>NFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston</td>
<td>506</td>
<td>2.42 ± 0.050</td>
<td>2.43 ± 0.15</td>
<td>2.46 ± 0.25</td>
<td>2.41 ± 0.25</td>
<td>2.37 ± 0.24</td>
<td>2.49 ± 0.11</td>
<td>2.48 ± 0.11</td>
</tr>
<tr>
<td>Concrete</td>
<td>1030</td>
<td>3.29 ± 0.02</td>
<td>3.04 ± 0.17</td>
<td>3.04 ± 0.09</td>
<td>3.06 ± 0.18</td>
<td>3.03 ± 0.11</td>
<td>3.09 ± 0.08</td>
<td>3.03 ± 0.13</td>
</tr>
<tr>
<td>Energy</td>
<td>768</td>
<td>1.06 ± 0.09</td>
<td>0.60 ± 0.45</td>
<td>1.99 ± 0.09</td>
<td>1.38 ± 0.22</td>
<td>0.66 ± 0.17</td>
<td>1.04 ± 0.09</td>
<td>1.21 ± 0.08</td>
</tr>
<tr>
<td>Kin8nm</td>
<td>8192</td>
<td>-0.99 ± 0.01</td>
<td>-0.49 ± 0.02</td>
<td>-0.95 ± 0.03</td>
<td>-1.20 ± 0.02</td>
<td>-1.11 ± 0.03</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Naval</td>
<td>11934</td>
<td>-6.54 ± 0.03</td>
<td>-5.34 ± 0.04</td>
<td>-3.80 ± 0.05</td>
<td>-5.63 ± 0.05</td>
<td>-4.98 ± 0.02</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Power</td>
<td>9568</td>
<td>2.85 ± 0.005</td>
<td>2.79 ± 0.11</td>
<td>2.80 ± 0.05</td>
<td>2.79 ± 0.04</td>
<td>2.81 ± 0.05</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Protein</td>
<td>45730</td>
<td>2.63 ± 0.006</td>
<td>2.81 ± 0.03</td>
<td>2.89 ± 0.01</td>
<td>2.83 ± 0.02</td>
<td>2.89 ± 0.02</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Wine</td>
<td>1588</td>
<td>0.67 ± 0.028</td>
<td>0.91 ± 0.06</td>
<td>0.93 ± 0.06</td>
<td>0.94 ± 0.12</td>
<td>0.95 ± 0.06</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Yacht</td>
<td>308</td>
<td>0.004 ± 0.046</td>
<td><strong>0.20 ± 0.26</strong></td>
<td>1.55 ± 0.12</td>
<td>1.18 ± 0.21</td>
<td>0.10 ± 0.26</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

The 2 CPDs (dashed and solid line) correspond to 2 picked observations in the respective data set.
Summary and outlook

- Transformation models allow to model highly flexible outcome \( \rightarrow \) extremely high prediction performance
- Any kind of input data and NN architectures can be integrated
- In the mean time we have extended the approach
  - to ordinal outcomes
  - to provide interpretable model parameters

Maximal flexibility but not interpretable:

\[
h(\tilde{y} | X = x) = \sum_{k=1}^{M} \vartheta_k(x) \cdot B e_k(\tilde{y})
\]

Interpretable but less flexible:

\[
h(y_k | x) = \vartheta_k - \beta \cdot x
\]

If \( x \) increases by one unit, the odds for a higher class increases by \( e^\beta \)