Robotics: Enter deep learning
towards end-to-end learning in robotics

Patrick van der Smagt
Biomimetic Robotics and Machine Learning lab — brml.org

Informatics, TUM (Technische Universität München)
fortiss, TUM Associate Institute
goal-oriented task solving: classical approach

- task-oriented sensor / vision model
- task-oriented planning model
- feedback error control in configuration space

Urbanek / Albu-Schäffer / van der Smagt, IROS 2004
goal-oriented task solving: servoing

- task-oriented vision model
- execute task in sensor domain
- feedback error control in sensor space
goal-oriented task solving: enter reinforcement learning

- states $s$, actions $a$, rewards $r$

- when the agent observes a state $s_t$ it chooses an action $a_t$ leading to a reward $r_t$ and state $s_{t+1}$

- task: learn a policy $\pi : S \rightarrow A$ to maximise $E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ...]$

A specific sensor model is typically used to reduce the dimensionality of $S$ and linearise it in control space.
next topic: a short history of neural networks (old stuff)

- 1960's: linear perceptron (Rosenblatt et al)
- 1969: the perceptron cannot do XOR (Minsky & Papert)
- 1970's–1980's: nonlinear neural networks with back-propagation (Linnainmaa; Dreyfus; Werbos; Rumelhart)
- early 1990s: one hidden layer suffices to represent any (Borel-measurable) function
- mid 1990's: neural networks can't do everything / don't generalise / too slow
- mid 1990's: Enter SVM
- 1995–2000: SVMs are too expensive / slow because of having too many Support Vectors
a short history of neural networks (new stuff)

- 2000–: probabilistic models for machine learning

- 2006: **deep neural networks**, trained with RBM + back-propagation

- 2009: deep NNs can also be trained by **good computing power** (GPUs) and having **many data**

- 2012: **dropout** prevents overfitting

- 2012–: cNNs beat many vision benchmarks

- 2012–: recurrent neural networks increasingly stable

- 2013–: **probabilistic** neural networks (variance propagation; variational autoencoder; ...)

- 2014–: end-to-end learning applications in robotics

- 2016–: Bayesian neural networks solve the requirement of needing many data
representing data dependencies with neural networks

\[ y = s(W_2 h) \]

\[ h = s(W_1 x), \text{ } s \text{ is some nonlinear function} \]

\[ x \]

\[ W_1 \text{ and } W_2 \text{ are found by back-propagating the loss } (y-z)^2 \]
representing data dependencies with **deep** neural networks
representing data dependencies with deep neural networks

\[
y = s(W_4 h_3) \\
h_3 = s(W_3 h_2) \\
h_2 = s(W_2 h_1) \\
h_1 = s(W_1 x) \\
x
\]

a human brain with $10^{11}$ neurons and $\sim 10^{12}$ axons lives $3 \cdot 10^9$ seconds

keynote speech held at the 2015 IROS conference in Hamburg, Germany
representing data with recurrent neural networks

\[
\begin{align*}
    y &= s(W_4 h_3) \\
    h_3 &= s(W_3 h_2 + V_3 h_3) \\
    h_2 &= s(W_2 h_1 + V_2 h_2) \\
    h_1 &= s(W_1 x + V_1 h_1) \\
    x
\end{align*}
\]
representing **data** with **recurrent** neural networks

\[
E(y), \ Var(y)
\]

\[
h_3 = s(W_3 h_2 + V_3 h_3)
\]

\[
h_2 = s(W_2 h_1 + V_2 h_2)
\]

\[
h_1 = s(W_1 x + V_1 h_1)
\]

\[
x
\]

e.g., Varprop: Justin Bayer, Christian Osendorfer, Sebastian Urban, Patrick van der Smagt (2013)
autoencoder

- deep neural network
- unsupervised learning

low-dimensional manifold
"latent space"
movement reconstruction using deep autoencoder with DMPs

\[ g_\theta^{(1)}(\cdots g_\theta^{(n)}\cdots) \]

\[ g_\theta^{(2)}(\cdots g_\theta^{(n)}\cdots) \]

\[ \tilde{y} \]

\[ \text{DMP} \]

\[ \tau \ddot{y} = \alpha_z(\beta_z(g - y) - \dot{y}) + f \]

\[ h_\theta^{(n)}(\cdots h_\theta^{(2)}\cdots) \]

\[ h_\theta^{(1)} \]

\[ X \]

\[ y \] "latent space"

keynote speech held at the 2015 IROS conference in Hamburg, Germany

Nutan Chen & Justin Bayer & Sebastian Urban & Patrick van der Smagt, Humanoids 2015
VAE: variational autoencoder

- deep neural network
- unsupervised learning
- loss = reconstruction - $KL(latents \parallel input)$
- input = $x$
  output = $E(x), \ Var(x)$
- "nonlinear PCA"

$KL(P \parallel Q)$ expresses the distance between two distributions

Durk Kingma and Max Welling, 2013
Kullback-Leibler-divergence

- $KL(P \parallel Q)$ expresses the distance between two distributions
- $KL(P \parallel Q)$ is the information lost when $Q$ is used to approximate $P$
- typically, $P$ is the real data, while $Q$ is the model

$$KL(P \parallel Q) = \int p(x) \log \frac{p(x)}{q(x)} \, dx$$
convolutional neural network
learning to grasp example: CNN for visual data

convolutional NN

Deep Learning for Detecting Robotic Grasps

Ian Lenz, Honglak Lee, Ashutosh Saxena (Cornell, 2013)

learning based on human "teacher"
(Cornell grasping dataset)
a random literature walk

- *Autonomous Off-Road Vehicle Control using End-to-End Learning*, 2005: map images to human steering (LeCun group)

- *Deep Learning for Detecting Robotic Grasps*, 2013 (Ian Lenz et al)


- *Human-level control through deep reinforcement learning*, Nature, 2015 (Google Deepmind)

- *Embed to control: A Locally Linear Latent Dynamics Model for Control from Raw Images*, arXiv, 2015 (Uni Freiburg / Riedmiller group)
end-to-end learning example: CNN for visual data with RL

trains unsupervised on raw images

- determine distance to rack to place hanger
- insertion task
- hammer task
- cap screwing task

despite visual clutter

Sergey Levine, Chelsea Finn, Trevor Darrell, Pieter Abbeel (CMU) (2 IROS presentations)
end-to-end learning example: optimal control in latent space

- directly learn from images (simulation)
- optimal control in latent space
- training is fully unsupervised

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker (U Freiburg), Martin Riedmiller (Google Deepmind)
balancing a pole using a tactile sensor:
Relative Entropy Policy Search using VAE features

3 Neural Network Control
As a first test if the unsupervised trained features are suitable for controlling the robot we first tried a very simple model. For this control task we used a neural network modelling the system dynamics as a one step predictor. This is done by using the state of the robot together with the current action as input for the neural network and training it to predict the next state. Actions are created by evaluating the neural network for all possible actions from a discrete set. The action chosen for controlling the robot is the one where the predicted next state is the best in terms of the reward function. We chose the reward function to be maximal at zero in the latent tactile space. This zero position corresponds to an angle close at the centre position in angular sensor space. The results using this method can be seen in 2. The plot shows the average reward over 10 experiments. Training the model is performed after each of the 30 rollouts during one experiment. As shown in Figure 2 the reward is steadily increasing until reaches the maximum of 1.0 at the end of each experiment.

4 Relative Entropy Policy Search
We also used Relative Entropy Policy Search (REPS) for controlling the robot. This algorithm will be used for most future tasks. One of the differences to the previously used one-step predictor is the optimization of a cumulative reward instead of a single evaluation. When using this type of algorithm we are now able to define the reward function in terms of external sensor values e.g. in our case the angular sensor value. For the experiments presented in Table 1 the reward function was defined to have a maximum at 0 degree. The first average reward shown in Table 1 is the reward for the initial random movement used for training in the next iteration. After one iteration the robot is able to move the pole to the centre position and increasing the average reward to a high level and keeps high after several iterations each followed by retraining the reinforcement model.

<table>
<thead>
<tr>
<th>iteration</th>
<th>random exploration</th>
<th>avg. reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>-35.4252</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>-11.9003</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>-24.6761</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>-15.6429</td>
</tr>
</tbody>
</table>

Table 1: Average reward for robot controlled with REPS.

Herke van Hoof & Nutan Chen & Maximilian Karl / Patrick van der Smagt & Jan Peters, 2015 (not yet published)
how to read braille with a BioTac?

**Gruppe 1**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

**Gruppe 2**

<table>
<thead>
<tr>
<th>k</th>
<th>l</th>
<th>m</th>
<th>n</th>
<th>o</th>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

**Gruppe 3**

<table>
<thead>
<tr>
<th>u</th>
<th>v</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>β</th>
<th>st</th>
</tr>
</thead>
<tbody>
<tr>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

**Gruppe 4**

<table>
<thead>
<tr>
<th>au</th>
<th>eu</th>
<th>ei</th>
<th>ch</th>
<th>sch</th>
<th>ü</th>
<th>ö</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

**Gruppe 5**

<table>
<thead>
<tr>
<th>su</th>
<th>ä</th>
<th>ie</th>
<th>Zahlt.</th>
<th>Großt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>●</td>
<td>●</td>
<td>●</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

**Gruppe 6**

<table>
<thead>
<tr>
<th>,</th>
<th>;</th>
<th>:</th>
<th>?</th>
<th>!</th>
<th>(</th>
<th>)</th>
<th>&quot;</th>
<th>&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>

---

**Figure: BioTac Sensor**

<table>
<thead>
<tr>
<th>No. of Taxels</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Taxel Data</td>
<td>Resistive</td>
</tr>
<tr>
<td>DC Pressure Range</td>
<td>0–100 kPa</td>
</tr>
<tr>
<td>DC Temp. Range</td>
<td>0–75°C</td>
</tr>
<tr>
<td>AC Pressure spectrum</td>
<td>10–1040 Hz</td>
</tr>
<tr>
<td>AC Temp. spectrum</td>
<td>0.45–22.6 Hz</td>
</tr>
</tbody>
</table>

---

keynote speech held at the 2015 IROS conference in Hamburg, Germany
A keynote speech held at the 2015 IROS conference in Hamburg, Germany.

BioTac in a supporting "wrist".
tasks
1. find the tactile data
2. follow a line

empowerment
Polani et al, 2008–

\[ C(s) := \max_{p(a|s)} \int p(a|s) \int p(s'|s,a) \ln \frac{p(s'|s,a)}{p(s'|s)} ds' da \]

probalistic neural network (Varprop) on VAE latents

neural network on VAE latents

KL-divergence between \( s'|s, a \) and \( s'|s \) maximises information gain

trick: we use two VAEs to compute i.i.d. representations of \( s \) and \( a \) no more sampling!

\( a \) brings the system from state \( s \) to \( s' \)

Maximilian Karl, Justin Bayer, Patrick van der Smagt: Efficient Empowerment, arXiv 2015

example: empowerment on pole balancing finds the "most interesting" region for the pole to be in

keynote speech held at the 2015 IROS conference in Hamburg, Germany
tasks
1. find the tactile data
2. follow a line

empowerment

information found on the page after 15' exploration
(and "a bit"of training 4 deep neural networks)
tasks

1. find the tactile data
2. follow a line
3. translate dots to characters

moving on the line 5mm too low

moving on the line 5mm too high
tasks

2. follow a line
3. translate dots to characters
4. translate characters to words
tasks
1. find the tactile data
2. follow a line
3. translate dots to characters
4. translate characters to words

VAE encoding of tactile data in latent space
(using segmented characters)

latents fed into a deep neural network
FD-DRNN encoding of 40 time steps of tactile data

Fast-Dropout Deep Recurrent Neural Network

- 91% dot accuracy
- 59% character accuracy
keynote speech held at the 2015 IROS conference in Hamburg, Germany

tasks

1. find the tactile data
2. follow a line
3. translate dots to characters
4. translate characters to words

<table>
<thead>
<tr>
<th>truth</th>
<th>1st guess</th>
<th>2nd guess</th>
<th>3rd guess</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>S</td>
<td>P</td>
<td>I</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
<td>A</td>
<td>H</td>
</tr>
<tr>
<td>R</td>
<td>M</td>
<td>O</td>
<td>L</td>
</tr>
<tr>
<td>M</td>
<td>A</td>
<td>D</td>
<td>K</td>
</tr>
<tr>
<td>A</td>
<td>N</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>N</td>
<td>E</td>
<td>O</td>
<td>Q</td>
</tr>
<tr>
<td>E</td>
<td>N</td>
<td>J</td>
<td>Q</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>J</td>
<td>Q</td>
</tr>
</tbody>
</table>

L | U | D | W | I | G
---|---|---|---|---|---
L | U | N | CH| I | G
R | Z | D | SCH| J | H
; | V | O | D | S | F
P | A | R | T | N | E | R
N | B | O | T | N | E | R
Q | A | Z | S | O | SCH| O
M | H | R | Q | Y | A | L