Deep Learning, Echo State Networks and the Edge of Chaos

Artificial Intelligence Seminar 1. 11. 2016
Filip Matzner, František Mráz
Neuron

**Biological**

**Artificial**

\[ y_j = \varphi \left( \sum_{i=1}^{N} w_{ij} x_i + b_j \right) \]
Network Architectures

biological neural networks are recurrent
artificial recurrent networks are hard to train
⟹ feed-forward networks receive more attention
Feed-Forward Networks (Pre-2010)

shallow = no more than a single hidden layer

sigmoidal activation function

\[ y = \frac{1}{1 + e^{-x}} \]

mean squared error cost function

\[ C_x(w, b) = \frac{\|y(x) - a\|^2}{2} \]
Feed-Forward Supervised Training

gradient descent

a method for function minimization

idea: put a ball on the cost function surface and let it roll down

it is too simple, where is the hatch?

in calculating the derivatives for each weight

\[ \nabla C = \left( \frac{\partial C}{\partial w_{11}}, \frac{\partial C}{\partial w_{12}}, \ldots, \frac{\partial C}{\partial w_{jk}} \right) \]
Backpropagation

a gradient descent method specialized on neural networks

efficiently propagates the error gradient from the output layer to the input layer

calculates all the derivatives \( \nabla_C = \left( \frac{\partial C}{\partial w_{11}^{(1)}}, \frac{\partial C}{\partial w_{21}^{(2)}}, \ldots, \frac{\partial C}{\partial w_{jk}^{L}} \right) \) in a single pass

Summary: the equations of backpropagation

\[
\begin{align*}
\delta^L &= \nabla_a C \odot \sigma'(z^L) \quad \text{(BP1)} \\
\delta^l &= ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l) \quad \text{(BP2)} \\
\frac{\partial C}{\partial w_j^l} &= \delta^l_j \quad \text{(BP3)} \\
\frac{\partial C}{\partial w_{jk}^l} &= \delta_k^{l-1} \delta_j^l \quad \text{(BP4)}
\end{align*}
\]
Vanishing Gradient

why the networks cannot not be deeper?

imagine a simple network with three hidden layers

\[
\frac{\partial C}{\partial b_1} = \sigma'(z_1) \times w_2 \times \sigma'(z_2) \times w_3 \times \sigma'(z_3) \times w_4 \times \sigma'(z_4) \times \frac{\partial C}{\partial a_4}
\]

now plot the derivative of the sigmoid function

usually \(|w_j \sigma(z_j)| < \frac{1}{4}\), thus with each layer, the gradient exponentially decreases

problem first described by Bengio et al. [1994]
this simple description taken from Nielsen [2015]
Outline

- NEAT
- HyperNEAT
- neuroevolution
- backprop
- supervised
- reinforcement
- training
- shallow architectures
- pre-2010
- feed forward
- neural networks
- post-2010
- recurrent
Feed-Forward Neuroevolution

in the simplest possible scenario

consider the network weights to be a vector of real numbers

evolve the weights by the basic genetic algorithm

this does not work particularly well, specialized evolutionary algorithms exist
NEAT

one of the many existing neuroevolutionary algorithms
good for smaller networks, not as much for large ones
can evolve both feed-forward networks and recurrent networks
even though it is quite old, it provides a suitable baseline

Stanley and Miikkulainen [2002]
HyperNEAT

extension of NEAT, can evolve larger networks

good when the network contains a lot of regularities

Stanley et al. [2009]
Feed-Forward Networks (Post-2010)

deep = more than two hidden layers

GPU implementations (100x speedup)

ReLU activation function \( y = \max(0, x) \)
Cross-Entropy Cost Function

saturated neurons learn slowly with MSE even though their error might be the largest one

why? because of the shape of the sigmoid function

\[ \frac{\partial C}{\partial w} = (a - y)\sigma'(z) \]

Derivation \( \frac{\partial C}{\partial w_{jk}} \) at this point is small

it can be partially avoided using cross-entropy cost function

\[ C = -\frac{1}{n} \sum_{x_j} [y_j \ln a_j^L + (1 - y_j) \ln(1 - a_j^L)] \]
Softmax Activation

another way to address the learning slowdown, especially when combined with the cross-entropy cost function

emphasizes the neuron with the maximum activation, however, does not ignore the other neurons

\[ a_j^L = \frac{e^{z_j^L}}{\sum_k e^{z_k^L}} \]

can be thought of as a probabilistic distribution, because

\[ \sum_j a_j^L = \frac{\sum_j e^{z_j^L}}{\sum_k e^{z_k^L}} = 1 \]
Convolutional Layers

apply the same convolutional kernel to all the pixels in the image

a way to share the weights between neurons $\Rightarrow$ regularization

inspired by nature
Max Pooling

similar to the convolution but only takes the maximum of the perception field

a good way to subsample the image (i.e., fewer parameters to train)

forces more succinct image representation (i.e., compression)
Dropout

randomly disable some of the neurons in each backprop step

(a) Standard Neural Net
(b) After applying dropout.

Hinton et al. [2012]
Data Augmentation

enlarge the dataset by transforming the data so that they still make sense

horizontal flip, rotation, scale, color inversion, ...

reduces overfitting
Supervised Training

backprop variants, e.g.,
  RMSProp (unpublished, see Hinton Neural Networks on Coursera [2012])
  Adam (Kingma and Ba [2015])

the derivatives do not have to be calculated manually, your favourite deep learning framework does it for you
Neuroevolution

the networks are usually too large to be evolved purely by neuroevolution
not even a combination of neuroevolution and supervised training is usual

    a single training on the GPU usually takes hours or days
ILSVRC

- benchmark of image classification models
- ~1.5 million images
- 1000 classes

Russakovsky et al. [2015]
AlexNet

- the winner of ILSVRC image classification 2012
- 8 layers
- 15.3% top-5 error
- 60 million parameters, 2 GPUs
- the beginning of the neural-network-computer-vision era

Krizhevsky et al. [2012]
AlexNet

- One GPU evolved color filters, the other evolved black and white filters
AlexNet - Classification Results

Figure 4: (Left) Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). (Right) Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

Krizhevsky et al. [2012]
AlexNet - Image Similarity

Krizhevsky et al. [2012]

Previous best by Zezula et al. [2005]
GoogLeNet

- the winner of ILSVRC image classification 2014
- 22 layers
- 6.67% top-5 error
- Google

Szegedy et al. [2014]
Residual Networks

- the winner of ILSVRC image classification 2015
- 152 layers
- 4.49% top-5 error
- Microsoft research

example of a 34-layers residual network

Kaiming He et al. [2015]
Image Captioning

A man in a helmet skateboarding before an audience. Man riding on edge of an oval ramp with a skate board. A man riding a skateboard up the side of a wooden ramp. A man on a skateboard is doing a trick. A man is grinding a ramp on a skateboard.

Lebret et al. [2015]
Image Captioning

Lebret et al. [2015]
Recurrent Networks

can be unfolded through time and reduced to feed-forward networks

\[ a_i \xrightarrow{f} x_{i+1} \xrightarrow{g} y_{i+1} \]

implies training by backpropagation (through time)

problem: implicitly infinitely deep

⇒ the vanishing gradient is even more significant

partial solution - LSTM, GRU, Echo State Networks
Geometrical Representation of Exploding Gradient

consider the dynamical system
\[ x_t = w\sigma(x_t - 1) + b \]

Fig. 6 illustrates the error surface
\[ E_{50} = (\sigma(x_{50}) - 0.7)^2 \]

possible solution: limit the gradient norm

Figure 6. We plot the error surface of a single hidden unit recurrent network, highlighting the existence of high curvature walls. The solid lines depict standard trajectories that gradient descent might follow. Using dashed arrow the diagram shows what would happen if the gradients is rescaled to a fixed size when its norm is above a threshold.

Pascanu et al. [2013]
Outline
Long-Short Term Memory (LSTM)

avoid vanishing gradient

use a special neuron with a “memory”

⇒ able to capture long-term dependencies

de-facto standard in recurrent neural networks

Hochreiter and Schmidhuber [1997]
Long-Short Term Memory (LSTM)

Hochreiter and Schmidhuber [1997]
Gated Recurrent Unit (GRU)

slightly “simplified” version of LSTM

performance is comparable with LSTM

Cho et al. [2014], comparison with LSTM by Chung et al. [2014]
The fire brigade has arrived. Adenauer is in a tough spot. Waiting, bringing support and comfort to Commonwealth countries do.
Handwriting Generation

Type a message into the text box, and the network will try to write it out longhand (this paper explains how it works, source code is available here). Be patient, it can take a while!

Text --- up to 100 characters, lower case letters work best

Style --- either let the network choose a writing style at random or prime it with a real sequence to make it mimic that writer's style.
- Felt the breath away when they are
- He dismissed the idea
- prison welfare office compliment
- She looked closely as she
- and think of the idea in being adapted for
- random style

Bias --- increasing the bias makes the samples more legible but less diverse. Using a high bias and a priming sequence makes the network write in a neater version of the original style.

Samples

Speech Recognition

“Using a LSTM, we cut our transcription errors by 49%.”

-- Google Voice Blogspot [2015]


Graves et al. [2013]
Robot Control

Fig. 3. Minimally invasive knot-tying. (A) The knot-tying procedure starts with the needle and three grippers in this configuration. (B) Gripper 1 takes the needle, and the thread is fed manually to gripper 3. (C) The thread is pulled through the puncture, and (D) wound around gripper 2. (E) Gripper 2 grabs the thread between the puncture and gripper 3. (F) The knot is finished by pulling the end of the thread through the loop.

Mayer et al. [2008]
And more...

machine translation

image caption generation

...
Echo State Networks

what about a totally random network?

Jaeger [2001]
Biological Motivation

ESN’s do not seem to be biologically plausible
improve them by a different topology?

⇒ neuroevolution (HyperNEAT)
Neuroevolution

5 runs, 2000 generations, 150 individuals
the best network has only local connections

what about a locally connected random network?

⇒ fast neuroevolution alternative
Dynamic Systems

recurrant networks are dynamic systems

⇒ we can measure the amount of chaos (*Lyapunov exponent*)

Sprott [2015]
Chaos
Chaos

recurrent networks have multiple levels of chaos

relation between dynamics and performance

Bertschinger and Natschläger [2004], Boedecker et al. [2012]
Chaos

neuroevolution prefers to be close to the edge of chaos
Conclusion

neural networks represent a very strong artificial intelligence model

there has been a huge step forward in the last decade

recurrent networks still have a long way to go

neural networks are good in the same things as humans

neural networks are bad in the same things as humans
Questions?
Recommended Reading

1) Neural Networks and Deep Learning - Michael Nielsen
A well written online book covering the basic topics of feed-forward neural networks. Freely available.
http://neuralnetworksanddeeplearning.com/

2) ImageNet Classification with Deep Convolutional Neural Networks - Krizhevsky et al. [2012].
One of the first great successes of deep convolutional networks. Short and clear paper, however, it assumes the knowledge of backprop, max-pooling, dropout, etc.

A book in preparation, which is available on-line. It is written by a few of the best deep learning scientists and describes most of the up-to-date techniques.
http://www.deeplearningbook.org/

4) Deep Learning in Neural Networks: An Overview - Schmidhuber et al. [2014]
http://arxiv.org/pdf/1404.7828v4