Advanced Data Analysis and Machine Learning

Lecture: Deep Learning

Lasse Lensu

2015-11-06

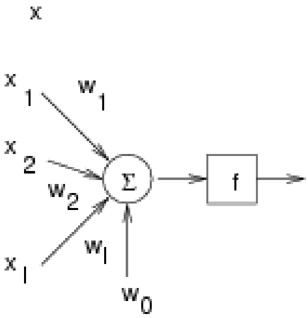
Outline

LUT
Lappeenranta
University of Technology

Perceptrons as network units

LUT Lappeenranta University of Technology

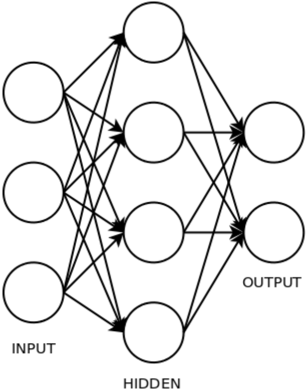
- Perceptrons are units of a neural network (NN).
- First, sum up the weighted input and threshold (bias).
- Second, map the sum through a nonlinear function f (*activation function*)
- A step function (hard limiter) is a commonly used nonlinearity, but sigmoidal functions are better for training.



Multilayer perceptrons

LUT Lappeenranta University of Technology

- A common type of artificial neural networks (ANNs).
- Several perceptrons are connected without loops, which is called *feed-forward* operation.
- Perceptrons are arranged in layers without connections within a layer.
- Backpropagation algorithm is used to propagate the error backwards in the network to re-adjust the perceptron weights in the case is undesired output.



Deep neural networks (DNNs)

LUT
Lappeenranta
University of Technology

- A deep neural network is a multi-layer neural network with several hidden layers.
- Significant improvements have been made to make the learning of large neural network efficient [2].
- Traditionally image analysis tasks have been solved by selecting suitable image features to represent the interesting image objects.
- Based on the image features, the object detection process can be realised by searching or classifying suitable object candidates, commonly in a supervised manner.
- Manual or "brute-force" selection of the appropriate image features can be avoided if the relevant features can be learned.
- This is the motivation for the convolutional neural networks (CNNs).

Neural network layer

LUT Lappeenranta University of Technology

- The features $\boldsymbol{h}$ of a fully connected layer are obtained by

$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}), \tag{1}$$

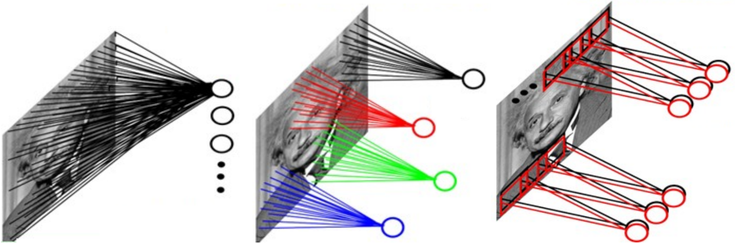
  where $f(\cdot)$ is the activation function, $\boldsymbol{W}^{H \times X}$ is the weight matrix, $\boldsymbol{b}^{H}$ is the bias vector, $\boldsymbol{x}^{X}$ is the input, X in the number of input units and H is the number of hidden units.
- Convolution $*$ is also a linear operation (cf. matrix multiplication in Eq. 1). If we denote the k-th feature detector as $\boldsymbol{W}_k$ and the corresponding bias as b_k , the k-th feature map is obtained as follows:

$$\boldsymbol{h}_k = g(\boldsymbol{W}_k * \boldsymbol{x} + b_k). \tag{2}$$

Network architectures

LUT Lappeenranta University of Technology

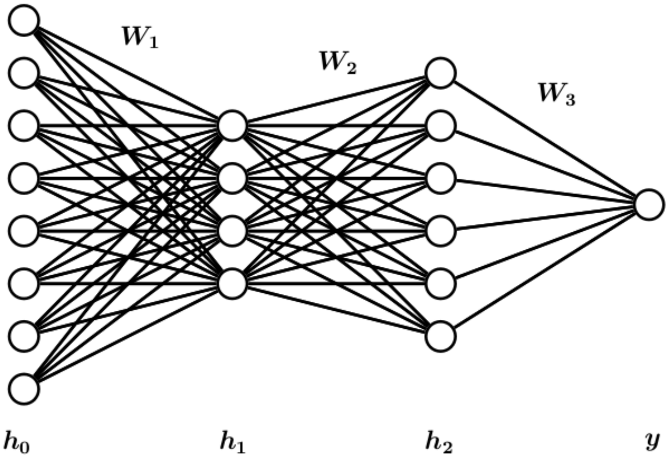
- Fully connected architecture
- Local receptive field architecture
- Local detectors with sliding window search



[1]

Feed-forward DNN

LUT
Lappeenranta
University of Technology



[1]

Dimensionality reduction

LUT Lappeenranta University of Technology

- A convolutional layer produces an overcomplete representation of its input (approx. k times where k is the number of convolutional filters).
- Max pooling: apply a max function for a given pooling window in the feature space.
- Pooling windows can be overlapping or not.
- In training, the error signals are propagated to the locations of the maximum values.
- The pooling adds a small translational invariance to the network architecture.
- An alternative to deterministic max pooling is stochastic pooling (also an additional regularisation method).

Training

- Backpropagation as a supervised training algorithm is used as with, for example, multilayer perceptron (MLP).
- Errors are propagated backwards in the network structure to re-adjust the weights.

Activation functions

LUT Lappeenranta University of Technology

- A set of linear functions can be replaced by just a single linear function, thus, a nonlinear activation function is needed to increase depth in the network.
- A sigmoid such as the standard logistic function

$$\text{logistic}(\boldsymbol{x}) = \frac{1}{1 + \exp(-\boldsymbol{x})} \tag{3}$$

is a monotonically increasing function. It is in wide use.

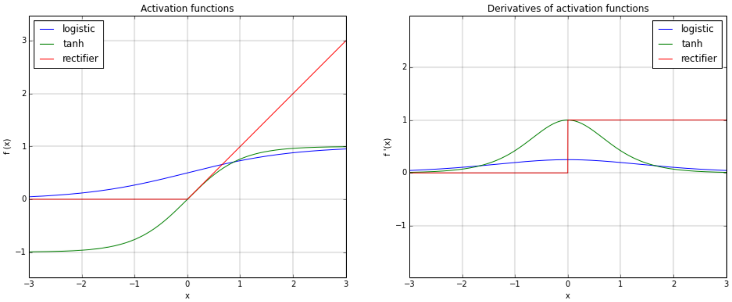
- A rectifier activation function

$$\text{rectifier}(\boldsymbol{x}) = \max(0, \boldsymbol{x}) \tag{4}$$

improves discriminative performance of convolutional networks.

- It introduces sparsity in the network (relatively small set of values are nonzero) and the function is linear when $\boldsymbol{x} > 0$.

Activation functions



[1]

Classification

- Classification can be performed by selecting the label based on the most probable class.
- However, it is useful in many cases to obtain the posterior probabilities for each class and use this information when the class likelihoods are close to each other.
- To obtain the probability for a class c, the following softmax function can be used:

$$P(\boldsymbol{Y} = c|\boldsymbol{x}) = \text{softmax}_c(\boldsymbol{x}) = \frac{\exp\left(\boldsymbol{W}_c\boldsymbol{x} + \boldsymbol{b}_c\right)}{\sum_j \exp\left(\boldsymbol{W}_j\boldsymbol{x} + \boldsymbol{b}_j\right)} \tag{5}$$

  where output units i have corresponding weight matrix $\boldsymbol{W}$ and bias vector $\boldsymbol{b}$.
- With softmax, a minimisable error function for learning at each layer is cross-entropy (CE) (desired output $\boldsymbol{y}_i$):

$$E_{\text{CE}} = -\sum_{i=1}^{n} \boldsymbol{y}_i \log\left(\hat{\boldsymbol{y}}_i\right) \tag{6}$$

Weight initialisation

LUT Lappeenranta University of Technology

- Optimisation of the network weights is difficult when the network has several hidden layers with nonlinear activation functions.
- If the initial weights are small, the error vanishes; If the initial weights are large, a poor solution is typically found.
- Two solutions have been proposed for the initialisation problem:
    - Generative pretraining (initial weights are close to a good solution)
    - Rectifier activation function

Preventing overfitting

LUT
Lappeenranta
University of Technology

- Deep neural networks rely on the number of parameters and nonlinear activation functions in a deep network, as well as on the overcomplete representation of the data.
- For example, with CNNs the purpose is to learn the relevant features, not just to memorise the input.
- To avoid overfitting, proper validation must be used to select the network with best generalisation capability.
- To achieve this, regularisation towards a sparse representation is needed (the whole network participates in learning the input-to-output mapping, not just a few units).

Preventing overfitting

LUT Lappeenranta University of Technology

- Generative pretraining can be used to initialise the weights (so that they are close to a good solution), after which it is possible fine-tune the weights discriminatively. Since the carrying idea is the ability to generate the data, the weights are strongly regularised.
- Dropout prevents overfitting: in training, randomly turn off network units and their connections with probability p, whereas in testing (model averaging), multiply the weights by the probability p.
- Weight-decay is commonly used to prevent weights from growing arbitrarily large. By replacing L2 weight-decay with L1 weight-decay introduces sparsity into the model.

Training data

LUT
Lappeenranta
University of Technology

- A large number of parameters requires a significant amount of data for training.
- To achieve good generalisation capability and invariance properties, data can be augmented.
- Making small variations to the training data can be used improve the learning of desired properties.

Optimising the weights

LUT Lappeenranta University of Technology

- The standard gradient descent (GD) is an iterative optimisation method making use of the gradient of the error function to be minimised.
- The update rule for GD is as follows:

$$\theta_{t+1} = \theta_t - \rho \nabla_{\theta_t} f(\boldsymbol{x}; \theta_t) \tag{7}$$

  where θ is the set of parameters, f is the function to be minimised, ρ is the learning rate, ∇_{θ_t} is the gradient.
- Selecting a suitable learning rate is difficult because the error function is rarely convex.
- Exponential decay, or scheduled learning rate and momentum optimisation can be used to avoid the problem.
- With a huge amount of data, it is infeasible to compute the gradient: the data can be divided into mini-batches and stochastic gradient descent (SGD) can be applied to estimate the gradient from the mini-batches.

Summary

- A deep neural network is a multi-layer neural network with several hidden layers.
- The architectures related to CNNs include fully connected architecture, local receptive field architecture and local detectors with sliding window search.
- Overfitting can be efficiently avoided by using proper validation and regularisation to realise a sparse representation.

References

Mikko Haavisto.
Pretraining convolutional neural networks for visual recognition tasks.
Master's thesis manuscript, 2015.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh.
A fast learning algorithm for deep belief nets.
Neural Comput., 18(7):1527–1554, 2006.