Introduction to Machine Learning

Chapter 3: Deep Learning- Multi-class Classification

Bernd Bischl

Department of Statistics – LMU Munich

Winter term 2021

Single Hidden Layer Networks for Multi-Class Classification

MULTI-CLASS CLASSIFICATION

- We have only considered regression and binary classification problems so far.
- How can we get a neural network to perform multiclass classification?

MULTI-CLASS CLASSIFICATION

- The first step is to add additional neurons to the output layer.
- Each neuron in the layer will represent a specific class (number of neurons in the output layer = number of classes).

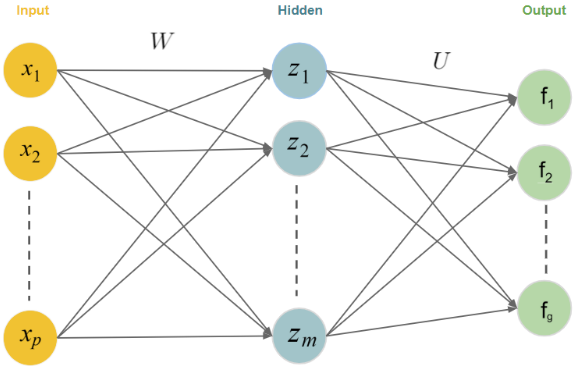


Figure: Structure of a single hidden layer, feed-forward neural network for g-class classification problems (bias term omitted).

MULTI-CLASS CLASSIFICATION

Notation:

- For g-class classification, g output units:

$$\mathbf{f} = (f_1, \ldots, f_g)$$

- m hidden neurons $z_1, \ldots, z_m$, with

$$z_j = \sigma(\mathbf{W}_j^\top \mathbf{x}), \quad j = 1, \ldots, m.$$

- Compute linear combinations of derived features z:

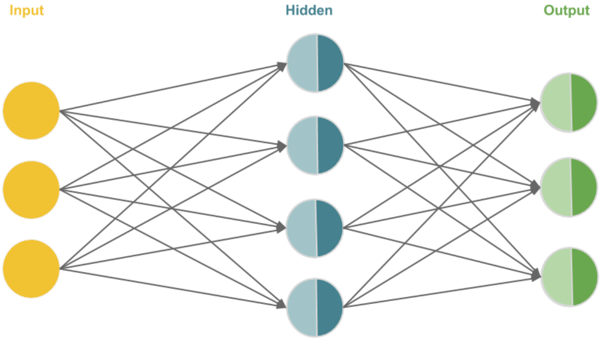
$$f_{in,k} = \boldsymbol{U}_k^\top \mathbf{z}, \quad \mathbf{z} = (z_1, \ldots, z_m)^\top, \quad k = 1, \ldots, g$$

MULTI-CLASS CLASSIFICATION

- The second step is to apply a softmax activation function to the output layer.
- This gives us a probability distribution over g different possible classes:
$$f_{out,k} = \tau_k(f_{in,k}) = \frac{\exp(f_{in,k})}{\sum_{k'=1}^{g} \exp(f_{in,k'})}$$
- This is the same transformation used in softmax regression!
- Derivative $\frac{\delta \tau(\mathbf{f}_{in})}{\delta \mathbf{f}_{in}} = \text{diag}(\tau(\mathbf{f}_{in})) - \tau(\mathbf{f}_{in})\tau(\mathbf{f}_{in})^\top$
- It is a "smooth" approximation of the argmax operation, so $\tau((1, 1000, 2)^\top) \approx (0, 1, 0)^\top$ (picks out 2nd element!).

MULTI-CLASS CLASSIFICATION: EXAMPLE

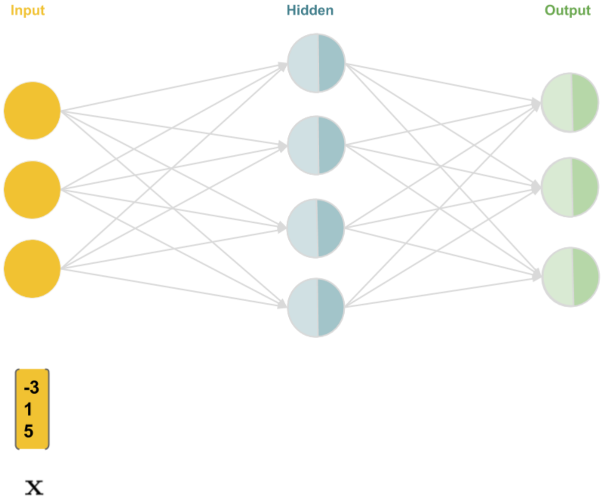
Forward pass (Hidden: Sigmoid, Output: Softmax).



$$\underbrace{\begin{pmatrix} 3 & -9 & 2 \\ 11 & -2 & 7 \\ -6 & 3 & -4 \\ 6 & -1 & 5 \end{pmatrix}}_{W^T} \underbrace{\begin{bmatrix} 5 \\ 2 \\ -1 \\ 1 \end{bmatrix}}_{\mathbf{b}} \qquad \underbrace{\begin{pmatrix} 3 & -12 & 8 & 1 \\ 2 & -3 & 9 & 1 \\ -5 & 1 & -1 & 7 \end{pmatrix}}_{U^T} \underbrace{\begin{bmatrix} 6 \\ 0 \\ -8 \end{bmatrix}}_{\mathbf{c}}$$

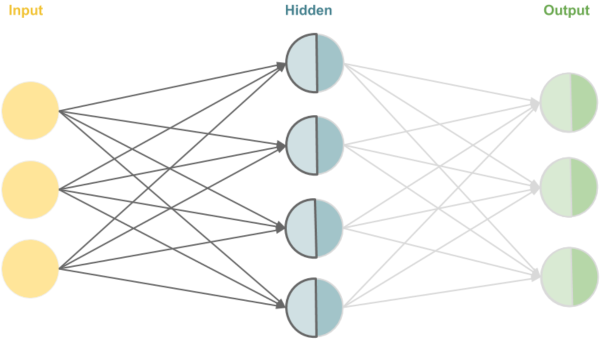
MULTI-CLASS CLASSIFICATION: EXAMPLE

Forward pass (Hidden: Sigmoid, Output: Softmax).



MULTI-CLASS CLASSIFICATION: EXAMPLE

Forward pass (Hidden: Sigmoid, Output: Softmax).

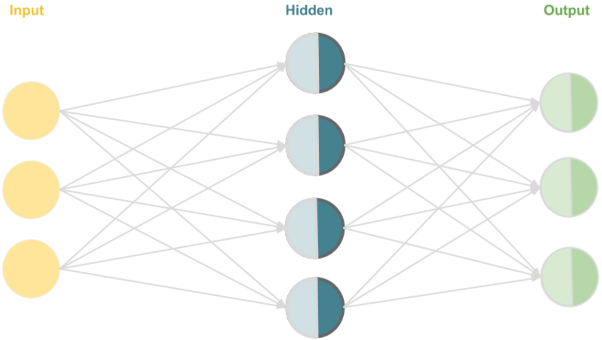


$$\underset{\mathbf{x}}{\begin{pmatrix} -3 \\ 1 \\ 5 \end{pmatrix}} \qquad \begin{pmatrix} (-3)*3 + 1*(-9) + 5*2 + 5 \\ (-3)*11 + 1*(-2) + 5*7 + 2 \\ (-3)*(-6) + 1*3 + 5*(-4) + (-1) \\ (-3)*6 + 1*(-1) + 5*5 + 1 \end{pmatrix}$$

$$\mathbf{x} \qquad \mathbf{z}_{in} = \mathbf{w}^T\mathbf{x} + \mathbf{b}$$

MULTI-CLASS CLASSIFICATION: EXAMPLE

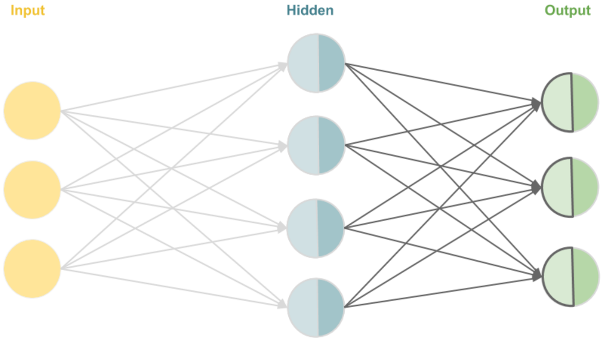
Forward pass (Hidden: Sigmoid, Output: Softmax).



$$\mathbf{x} = \begin{bmatrix} -3 \\ 1 \\ 5 \end{bmatrix} \qquad \mathbf{z}_{in} = \begin{bmatrix} -3 \\ 2 \\ 0 \\ 7 \end{bmatrix} \qquad \mathbf{z} = \mathbf{z}_{out} = \sigma(\mathbf{z}_{in}) = \begin{pmatrix} 1/(1+\exp(-(-3))) \\ 1/(1+\exp(-2)) \\ 1/(1+\exp(-0)) \\ 1/(1+\exp(-7)) \end{pmatrix}$$

MULTI-CLASS CLASSIFICATION: EXAMPLE

Forward pass (Hidden: Sigmoid, Output: Softmax).

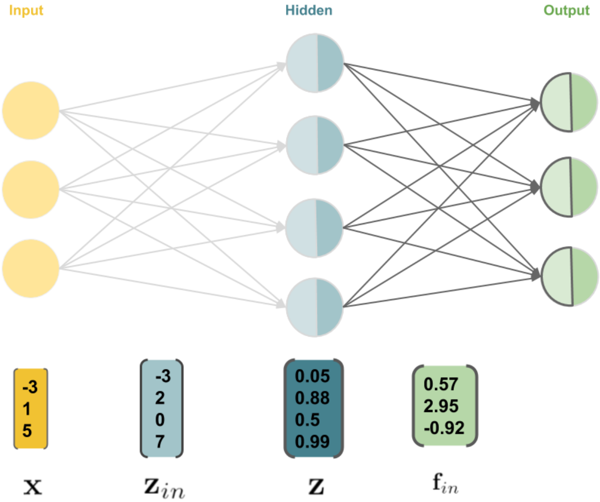


$$\mathbf{x} = \begin{pmatrix} -3 \\ 1 \\ 5 \end{pmatrix} \quad \mathbf{z}_{in} = \begin{pmatrix} -3 \\ 2 \\ 0 \\ 7 \end{pmatrix} \quad \mathbf{z} = \begin{pmatrix} 0.05 \\ 0.88 \\ 0.5 \\ 0.99 \end{pmatrix}$$

$$\mathbf{f}_{in} = U^T\mathbf{z} + \mathbf{c} = \begin{pmatrix} (0.05)*3 + (0.88)*(-12) + (0.5)*8 + (0.99)*1 + 6 \\ (0.05)*2 + (0.88)*(-3) + (0.5)*9 + (0.99)*1 + 0 \\ (0.05)*(-5) + (0.88)*1 + (0.5)*(-1) + (0.99)*7 + -8 \end{pmatrix}$$

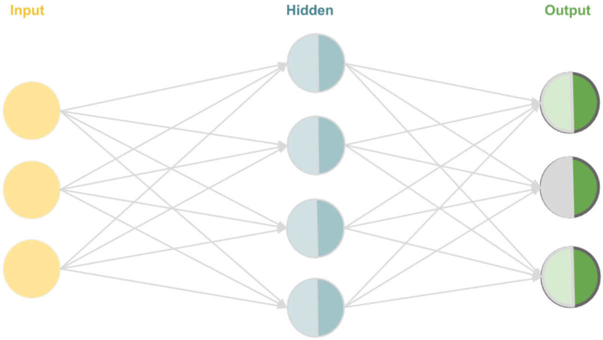
MULTI-CLASS CLASSIFICATION: EXAMPLE

Forward pass (Hidden: Sigmoid, Output: Softmax).



MULTI-CLASS CLASSIFICATION: EXAMPLE

Forward pass (Hidden: Sigmoid, Output: Softmax).

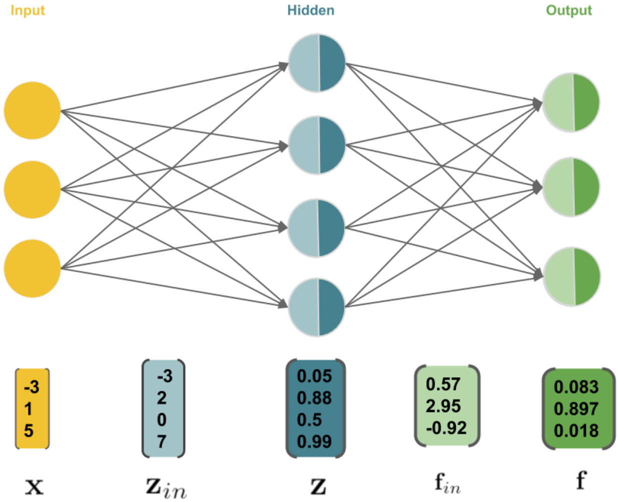


$$\begin{bmatrix} 0.57 \\ 2.95 \\ -0.92 \end{bmatrix} \quad \begin{pmatrix} \frac{\exp(0.57)}{\exp(0.57)+\exp(2.95)+\exp(-0.92)} \\ \frac{\exp(2.95)}{\exp(0.57)+\exp(2.95)+\exp(-0.92)} \\ \frac{\exp(-0.92)}{\exp(0.57)+\exp(2.95)+\exp(-0.92)} \end{pmatrix}$$

$$\mathbf{f}_{in} \qquad \mathbf{f} = \mathbf{f}_{out} = \tau(\mathbf{f}_{in})$$

MULTI-CLASS CLASSIFICATION: EXAMPLE

Forward pass (Hidden: Sigmoid, Output: Softmax).



SOFTMAX LOSS

- The loss function for a softmax classifier is

$$L(y, f(\mathbf{x})) = -\sum_{k=1}^{g} [y = k] \log \left(\frac{\exp(f_{in,k})}{\sum_{k'=1}^{g} \exp(f_{in,k'})} \right)$$

where $[y = k] = \begin{cases} 1 & \text{if } y = k \\ 0 & \text{otherwise} \end{cases}$.

- This is equivalent to the cross-entropy loss when the label vector $\boldsymbol{y}$ is one-hot coded (e.g. $\mathbf{y} = (0, 0, 1, 0)^\top$).
- Optimization: Again, there is no analytic solution.

SINGLE HIDDEN LAYER NETWORKS: SUMMARY

- We have seen that neural networks are far more flexible than linear models. Neural networks with a single hidden layer are able to approximate any continuous function.
- Yet, in reality, there is no way to make full use of the universal approximation property. The learning algorithm will usually not find the best possible model. At best it finds a locally optimal model.
- The XOR example showed us how neural networks extract features to transform the space and actually learn a kernel (learn a representation).
- Neural networks can perfectly fit noisy data. Thus, neural networks are endangered to over-fit. This is particularly true for a model with a huge hidden layer.
- Fitting neural networks with sigmoidal activation function is nothing else but fitting many weighted logistic regressions!