Applied Artificial Intelligence

Session 13: Feedforward Multilayer Neural Networks II

Fall 2018

NC State University
Lecturer: Dr. Behnam Kia

Course Website: https://appliedai.wordpress.ncsu.edu/

What will we learn in this session?

- Where those cost functions come from?
- Why type of problems can I solve with neural networks?
- New homework

The Problem Definition

- Imagine we have a set of observed data:

$$x_1, x_2, \ldots, x_m$$

- Assume we know which parametric probability distribution function ($\mathrm{p}(x|w)$) has produced the data, but we don't know the parameters (w?).

- Question: How to Estimate the Parameters?

The Problem Definition

- Imagine we have a set of observed data:

$$x_1, x_2, \ldots, x_m$$

- Assume we know which parametric probability distribution function $(\mathrm{p}(x|w))$ has produced the data, but we don't know the parameters (w?).

- Question: How to Estimate the Parameters

Common format: $p_w(x)$ is usually written as $\mathrm{p}(x|w)$ as well to indicate the dependence of probability values on the parameter *w*.

The Problem Definition

- Imagine we have a set of observed data:

$$x_1, x_2, \ldots, x_m$$

- Assume we know which parametric probability distribution function ($\mathrm{p}(x|w)$) has produced the data, but we don't know the parameters (w?).

- **Question: How to Estimate the Parameters**

Maximum Likelihood Estimation (MLE)

- MLE is an approach to estimate parameters.
- MLE: Choose parameters in a way that the observed data have the biggest probability.

$$w = \underset{w}{\operatorname{argmax}} P(Data|w)$$

- MLE tries to makes sense of observed data by making them more likely to be produced by the parametric model.

General Case

MLE says: $w_{MLE} = \underset{w}{\operatorname{argmax}}\, P(Data|w)$

$$P(Data|w) = P(x_1, x_2, \ldots, x_m|w)$$
$$= P(x_1|w)P(x_2|w) \ldots P(x_m|w) \qquad \text{(i.i.d assumption)}$$
$$= \prod_{i=1}^{m} P(x_i|w)$$

Example: Flipping a Coin

- Assume the observed data are the outcomes of flipping a coin:

- Data: H, T, T, H, H, H, T, H, H,…,T

- Question: What is parameter *p, (p=P(H))?*

Example: Flipping a Coin

- Assume the observed data are the outcomes of flipping a coin:
- Data: H, T, T, H, H, H, T, H, H,…,T
- Question: What is parameter *p?*
- MLE says: $p = \underset{p}{\operatorname{argmax}} P(Data|p)$

Log Likelihood for Numerical Stability

$$P(Data|w) = P(x_1, x_2, \dots, x_m|w)$$
$$= P(x_1|w)P(x_2|w) \dots P(x_m|w) \qquad (i.i.d \text{ assumption})$$
$$= \prod_{i=1}^{m} P(x_i|w)$$

$$\log(P(Data|w)) =$$
$$= \sum_{i=1}^{m} P(x_i|w)$$

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \log(P(Data|w))$$

MLE Example: Flipping a Coin Calculus-based Solution

$$P(Data|p) = P(H, T, T, \dots, T|p)$$
$$= P(H|p)P(T|p)P(T|p) \dots P(T|p) \quad (\textit{i.i.dassumption})$$
$$= p^k(1-p)^{n-k}$$

$$p_{MLE} = \underset{p}{\text{argmax}} \ (P(Data|p))$$

MLE Example: Flipping a Coin Calculus-based Solution

$$P(Data|p) = P(H,T,T,\dots,T|p)$$
$$= P(H|p)P(T|p)P(T|p)\dots P(T|p) \quad (\textit{i.i.dassumption})$$
$$= p^k(1-p)^{n-k}$$

$$p_{MLE} = \underset{p}{\text{argmax}}\ (P(Data|p))$$

- $\frac{\partial}{\partial p} P(Data|p) = 0$

MLE Example: Flipping a Coin Calculus-based Solution

$$P(Data|p) = P(H, T, T, \ldots, T|p)$$
$$= P(H|p)P(T|p)P(T|p) \ldots P(T|p) \quad \textit{(i.i.d assumption)}$$
$$= p^k(1-p)^{n-k}$$

$$p_{MLE} = \underset{p}{\text{argmax}} \ (P(Data|p))$$

$$\frac{\partial}{\partial p} p^k(1-p)^{n-k} = kp^{k-1}(1-p)^{n-k} - (n-k)p^k(1-p)^{n-k-1} = 0$$
$$k(1-p) \quad - (n-k)p = 0$$
$$k - np = 0$$
$$p = k/n$$

Supervised Machine Learning

- Imagine we have a set of training data:

$(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)$

- And our neural network is estimating $\mathrm{p}(y|x, w)$, but of course we don't know the optimal parameters (w?).

- Question: How to Estimate the Parameters?

Supervised Machine Learning

- Imagine we have a set of training data:
  $(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)$

- And our neural network is estimating $\mathrm{p}(y|x, w)$, but of course we don't know the optimal parameters (w?).

- Question: How to Estimate the Parameters?

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \log(P(y|x, w))$$

Supervised Machine Learning

- Imagine we have a set of training data:

$(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)$

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \log P\,(Y|X, w)$$
$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{m} \log P\left(y^i \middle| x^i, w\right)$$

Supervised Machine Learning
Logistic Regression

- Imagine we have a set of training data:

$(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)$

$$w_{MLE} = \underset{w}{\text{argmax}} \log P\,(Y|X, w)$$

$$= \underset{w}{\text{argmax}} \sum_{i=1}^{m} \log P\left(y^i \middle| x^i, w\right)$$

Reminder: in logistic regression:

$$\widehat{y^i}\ is\ P\left(output = 1 \middle| x^i, w\right)$$

1- $\widehat{y^i}$ is $P\left(output = 0 \middle| x^i, w\right)$

Supervised Machine Learning
Logistic Regression

- Imagine we have a set of training data:

$(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)$

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \log P\,(Y|X, w)$$

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{m} \log P\left(y^i \middle| x^i, w\right)$$

Reminder: in logistic regression:

$$\widehat{y^i} \; is \; P\left(output = 1 \middle| x^i, w\right)$$

1- $\widehat{y^i}$ is $P\left(output = 0 \middle| x^i, w\right)$

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \sum_{i=1}^{m} [y^i \; log\left(\widehat{y^i}\right) + (1 - y^i) \; log\left(1 - \widehat{y^i}\right)]$$

Supervised Machine Learning
Logistic Regression

- Imagine we have a set of training data:

$$(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)$$

$$w_{MLE} = \underset{w}{\text{argmax}} \log P\,(Y|X, w)$$

$$= \underset{w}{\text{argmax}} \sum_{i=1}^{m} \log P\left(y^i \middle| x^i, w\right)$$

Reminder: in logistic regression:

$$\widehat{y^i} \text{ is } P\left(output = 1 \middle| x^i, w\right)$$

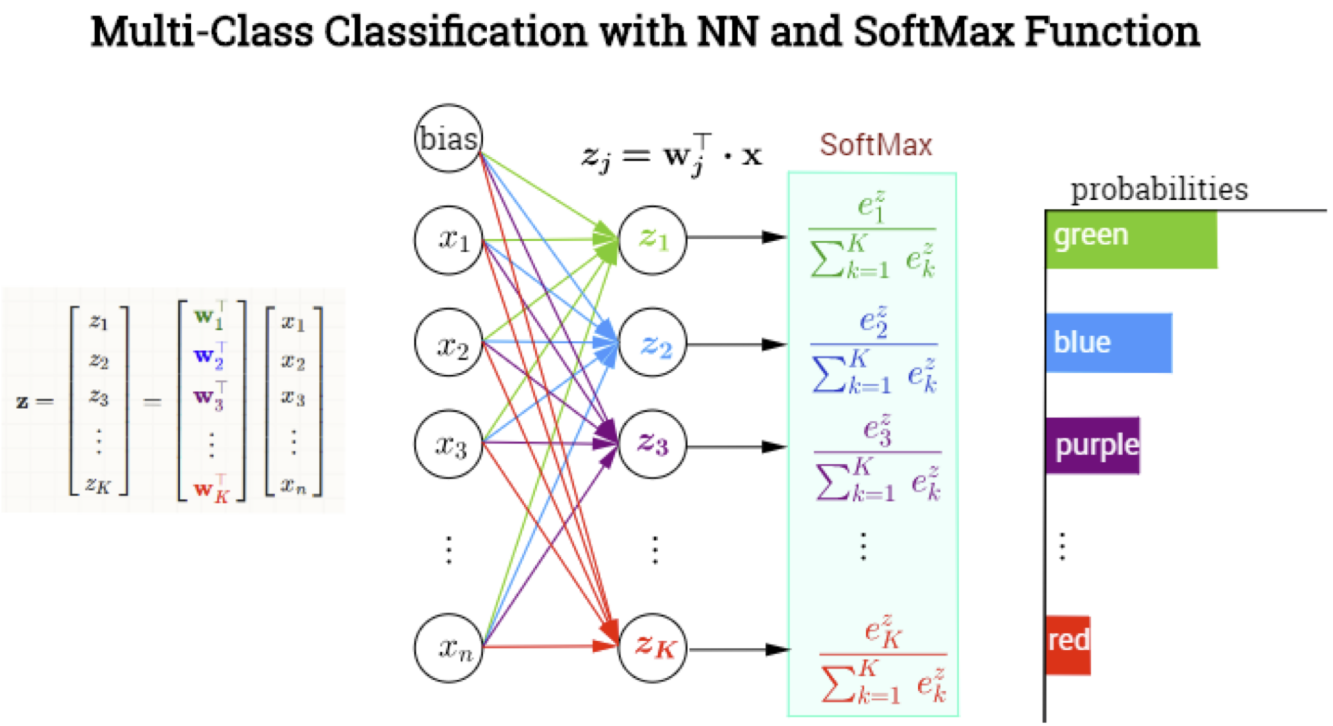
$$1\text{- } \widehat{y^i} \text{ is } P\left(output = 0 \middle| x^i, w\right)$$

$$w_{MLE} = \underset{w}{\text{argmax}} \sum_{i=1}^{m} [y^i log\left(\widehat{y^i}\right) + (1 - y^i)\, log\left(1 - \widehat{y^i}\right)]$$

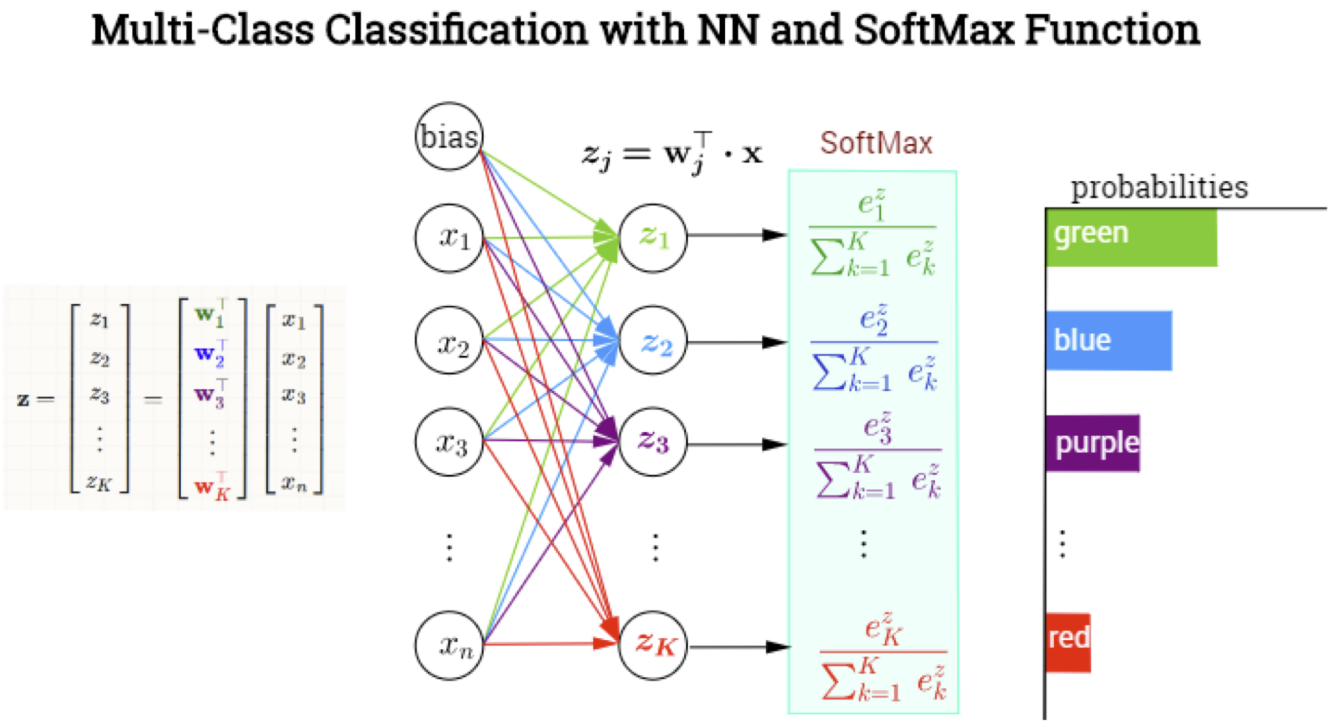
$$w_{optimal} = \underset{w}{\text{argmin}}\ \text{J(W)}$$

$$= \underset{w}{\text{argmin}} - \sum_{i=1}^{m} [y^i log\left(\widehat{y^i}\right) + (1 - y^i)\, log\left(1 - \widehat{y^i}\right)]$$

Classification with more than two categories



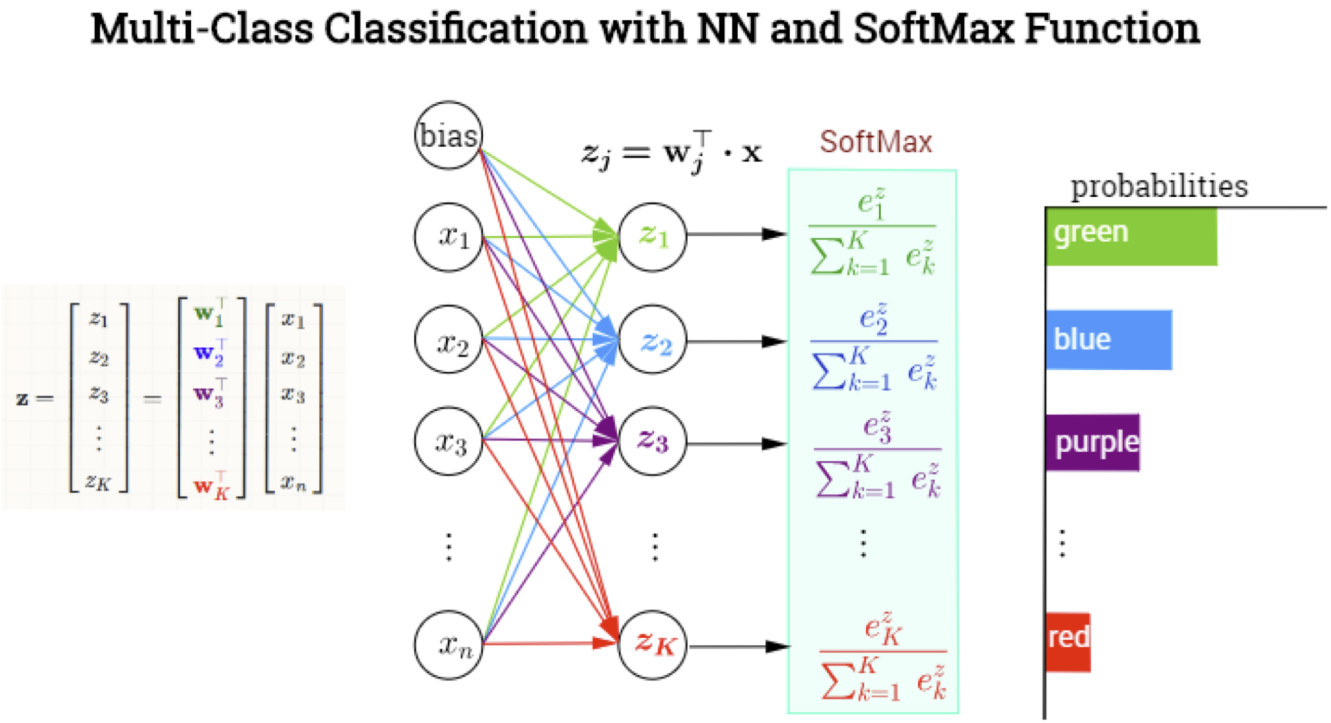
Classification with more than two categories



$$w_{MLE} = \underset{w}{\operatorname{argmax}} \log P(Y|X, w)$$
$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{m} \log P\left(y^i \middle| x^i, w\right)$$

$$J(W) = -\sum_{i=1}^{m} \log P\left(y^i \middle| x^i, w\right)$$

Classification with more than two categories



(X_1,Green)
(X_2,Purple)
.
.
.

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \log P\,(Y|X,w)$$
$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{m} \log P\left(y^i|x^i,w\right)$$

$$J(W) = -\sum_{i=1}^{m} \log P\left(y^i|x^i,w\right)$$

Cross entropy between model distribution and the training data

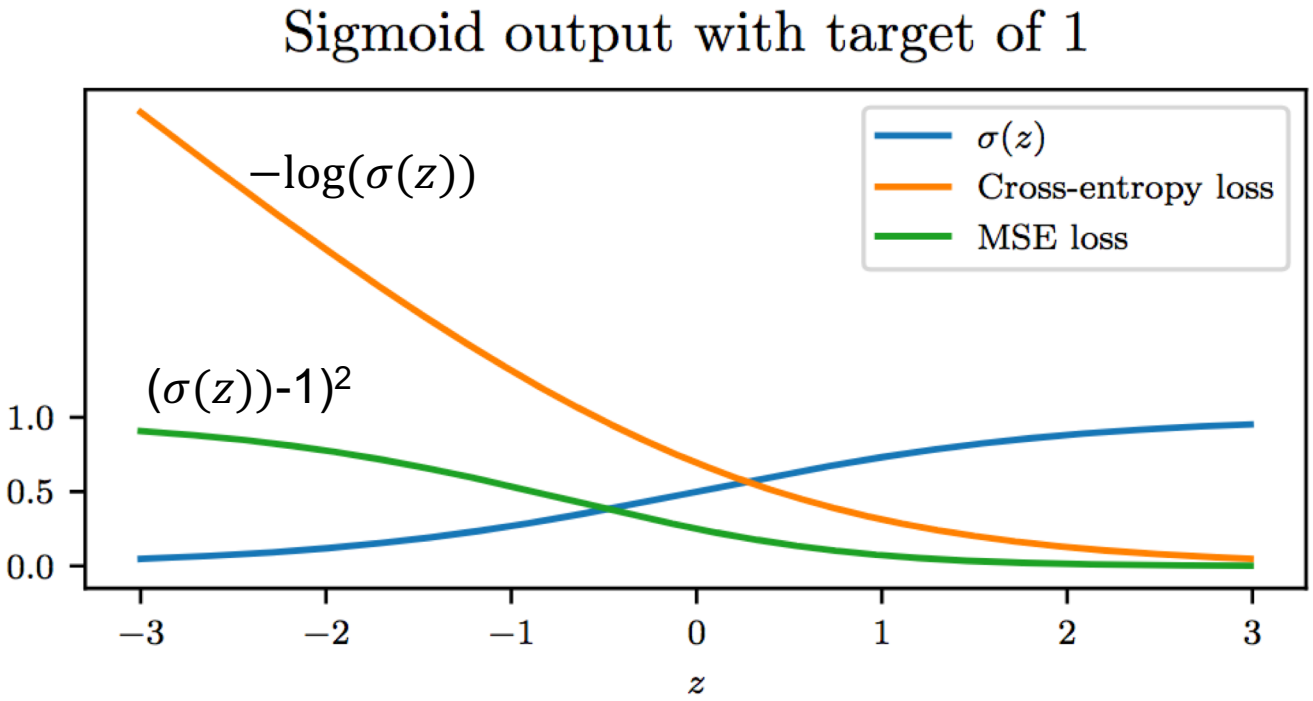
$$J(\boldsymbol{\theta}) = -\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\hat{p}_{\text{data}}} \log p_{\text{model}}(\boldsymbol{y} \mid \boldsymbol{x}).$$

Cross entropy between model distribution and the training data

$$J(\boldsymbol{\theta}) = -\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\hat{p}_{\text{data}}} \log p_{\text{model}}(\boldsymbol{y} \mid \boldsymbol{x}).$$

$$\sum_{i=1}^{m}$$

Output Type	Output Distribution	Output Layer	Cost Function
Binary	Bernoulli	Sigmoid	Binary cross-entropy
Discrete	Multinoulli	Softmax	Discrete cross-entropy
Continuous	Gaussian	Linear	Gaussian cross-entropy (MSE)
Continuous	Mixture of Gaussian	Mixture Density	Cross-entropy
Continuous	Arbitrary	See part III: GAN, VAE, FVBN	Various


Sigmoid output with target of 1
$-\log(\sigma(z))$
$(\sigma(z))$-1)^2
$\sigma(z)$
Cross-entropy loss
MSE loss
1.0
0.5
0.0
−3
−2
−1
0
1
2
3
z

Neural Networks: What they can do?

Brain is a universal learning machine

Visual Projections Routed to the Auditory Pathway in Ferrets: Receptive Fields of Visual Neurons in Primary Auditory Cortex

Anna W. Roe,[a] Sarah L. Pallas,[b] Young H. Kwon, and Mriganka Sur

Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

How does cortex that normally processes inputs from one sensory modality respond when provided with input from a different modality? We have addressed such a question with an experimental preparation in which retinal input is routed to the auditory pathway in ferrets. Following neonatal surgical manipulations, a specific population of retinal ganglion cells is induced to innervate the auditory thalamus and provides visual input to cells in auditory cortex (Sur et al., 1988). We have now examined in detail the visual response prop-

ulate nucleus, or MGN). Retinal afferents subsequently provide visual input to cells in primary auditory cortex (A1) of the "rewired" ferret (Sur et al., 1988) and establish a topographic visual map there (Roe et al., 1990a). We have now examined both qualitatively and quantitatively the physiological response properties of single visual units in A1 of rewired ferrets and compared them to the properties of cells in primary visual cortex (V1) of normal ferrets.

A preliminary report of these data has been published pre-

BrainPort® Vision Pro

The new BrainPort® Vision Pro is a 2nd generation oral electronic vision aid that provides electro-tactile stimulation to aid profoundly blind patients in orientation, mobility, and object recognition as an adjunctive device to other assistive methods such as the white cane or a guide dog.

BrainPort Vision Pro translates digital information from a wearable video camera into gentle electrical stimulation patterns on the surface of the tongue. Users feel moving bubble-like patterns on their tongue which they learn to interpret as the shape, size, location and motion of objects in their environment. Some users have described it as being able to “see with your tongue”.

Neural Networks

- One learning algorithm hypothesis
- Robotics, Speech Processing, Image Processing, NLP, etc, are not separate fields, and common learning algorithms can apply to all.