HW5

12011923 张旭东

Q1:

Given a set of independent observations of $\mathbf{x}$ and $\mathbf{t}$, $\mathbf{y}(\mathbf{x}, \mathbf{w})$ is the output of a neural network with input vector $\mathbf{x}$ and weight vector $\mathbf{w}$, so for each pair of $\mathbf{x}$ and $\mathbf{w}$, the error is

$$\mathbf{e} = \mathbf{t} - \mathbf{y}(\mathbf{x}, \mathbf{w}) \tag{1}$$

Because

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \mathcal{N}(\mathbf{t}|\mathbf{y}(\mathbf{x}, \mathbf{w}), \mathbf{\Sigma}) \tag{2}$$

So

$$p(\mathbf{e}|\mathbf{t}, \mathbf{x}, \mathbf{w}) = \mathcal{N}(\mathbf{e}|\mathbf{0}, \mathbf{\Sigma}) \tag{3}$$

For each pair of $\mathbf{x}$ and $\mathbf{t}$, the error function is

$$\mathbf{e}(\mathbf{w}, \mathbf{\Sigma}) = \frac{1}{\sqrt{2\pi|\mathbf{\Sigma}|}}\exp(-\mathbf{e}^T|\mathbf{\Sigma}^{-1}\mathbf{e}) \tag{4}$$

$$= \frac{1}{\sqrt{2\pi|\mathbf{\Sigma}|}}\exp(-(\mathbf{t} - \mathbf{y}(\mathbf{x}, \mathbf{w}))^T\mathbf{\Sigma}^{-1}(\mathbf{t} - \mathbf{y}(\mathbf{x}, \mathbf{w}))) \tag{5}$$

The total error function is

$$\mathbf{E}(\mathbf{w}, \mathbf{\Sigma}) = \prod_{n=1}^{N}\frac{1}{\sqrt{2\pi|\mathbf{\Sigma}|}}\exp(-(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))^T\mathbf{\Sigma}^{-1}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))) \tag{6}$$

The log likelihood function is given by

$$\ln L(\mathbf{w}, \mathbf{\Sigma}) = -\frac{N}{2}(\ln|\mathbf{\Sigma}| + K\ln(2\pi)) - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))^T\mathbf{\Sigma}^{-1}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w})) \tag{7}$$

K is the dimensionality of $\mathbf{y}$ and $\mathbf{t}$.

(a) if we assume that Σ is fixed and known, we can drop terms that are independent of $\mathbf{w}$ from `(6)`, and by changing the sign we get the error function:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))^T\mathbf{\Sigma}^{-1}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w})) \tag{8}$$

(b) if we consider maximizing `(6)` w.r.t. Σ, the terms that need to be kept are

$$-\frac{N}{2}\ln|\mathbf{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))^T\mathbf{\Sigma}^{-1}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w})) \tag{9}$$

By rewriting the second term we get

$$-\frac{N}{2}\ln|\mathbf{\Sigma}| - \frac{1}{2}Tr[\mathbf{\Sigma}^{-1}\sum_{n=1}^{N}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))^T] \tag{10}$$

We can maximize this by setting the derivative w.r.t. Σ^{-1} to zero, yielding

$$\mathbf{\Sigma} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))(\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}))^T \tag{11}$$

Thus the optimal value for Σ depends on $\mathbf{w}$ through $\mathbf{y}(\mathbf{x}_n, \mathbf{w})$.

A possible way to address this mutual dependency between $\mathbf{w}$ and $\mathbf{\Sigma}$ when it comes to optimization, is to adopt an iterative scheme, alternating between updates of $\mathbf{w}$ and $\mathbf{\Sigma}$ until some convergence criterion is reached.

Q2:

For a network having a logistic sigmoid output activation function

$$y(\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x}) = \frac{1}{1 + \exp(\mathbf{w}^T\mathbf{x})} \in [0, 1] \tag{12}$$

For a network having an output $-1 \leq y(\mathbf{x}, \mathbf{w}) \leq 1$, the relationship between $y_{new}(\mathbf{x}, \mathbf{w})$ and $y_{old}(\mathbf{x}, \mathbf{w})$ is

$$y_{new}(\mathbf{x}, \mathbf{w}) = 2y_{old}(\mathbf{x}, \mathbf{w}) - 1 \in [-1, 1] \tag{13}$$

So

$$\begin{aligned} y_{new}(\mathbf{x}, \mathbf{w}) &= 2\sigma(\mathbf{w}^T\mathbf{x}) - 1 && (14)\\ &= \frac{2}{1 + \exp(\mathbf{w}^T\mathbf{x})} - 1 && (15)\\ &= \frac{1 - \exp(\mathbf{w}^T\mathbf{x})}{1 + \exp(\mathbf{w}^T\mathbf{x})} && (16)\\ &= \frac{\exp(\frac{1}{2}\mathbf{w}^T\mathbf{x}) - \exp(-\frac{1}{2}\mathbf{w}^T\mathbf{x})}{\exp(\frac{1}{2}\mathbf{w}^T\mathbf{x}) + \exp(-\frac{1}{2}\mathbf{w}^T\mathbf{x})} && (17)\\ &= \tanh(\frac{1}{2}\mathbf{w}^T\mathbf{x}) && (18) \end{aligned}$$

So the new activation function is $\tanh(\frac{a}{2})$.

The error function for the network having a logistic sigmoid output activation function is

$$E(\mathbf{w}) = -\sum_{n=1}^{N}[t_n \ln y_n + (1 - t_n)\ln(1 - y_n)] \tag{19}$$

From the formula `(13)`, we can apply the same transformation to y_n and t_n. So the error function of the new network is

$$E(\mathbf{w}) = -\sum_{n=1}^{N}[\frac{t_n + 1}{2}\ln\frac{1 + y_n}{2} + (1 - \frac{t_n + 1}{2})\ln(1 - \frac{y_n + 1}{2})] \tag{20}$$

Q3:

For the mixture density network model,

$$p(\mathbf{t}|\mathbf{x}) = \sum_{k=1}^{K}\pi_k(\mathbf{x})\mathcal{N}(\mathbf{t}|\mu_k(\mathbf{x}), \sigma_k^2(\mathbf{x})) \tag{21}$$

$$E[\mathbf{t}|\mathbf{x}] = \int \mathbf{t}p(\mathbf{t}|\mathbf{x})d\mathbf{t}$$

$$= \int \mathbf{t} \sum_{k=1}^{K} \pi_k(\mathbf{x}) \mathcal{N}(\mathbf{t}|\mu_k(\mathbf{x}), \sigma_k^2(\mathbf{x}))d\mathbf{t} \tag{22}$$

$$= \sum_{k=1}^{K} \pi_k(\mathbf{x}) \int \mathbf{t} \mathcal{N}(\mathbf{t}|\mu_k(\mathbf{x}), \sigma_k^2(\mathbf{x}))d\mathbf{t}$$

The integral $\int \mathbf{t} \mathcal{N}(\mathbf{t}|\mu_k(\mathbf{x}), \sigma_k^2(\mathbf{x}))d\mathbf{t}$ just gives the mean of the Gaussian which is $\mu_k(\mathbf{x})$, hence

$$E[\mathbf{t}|\mathbf{x}] = \int \mathbf{t}p(\mathbf{t}|\mathbf{x})d\mathbf{t} = \sum_{k=1}^{K} \pi_k(\mathbf{x}) \mu_k(\mathbf{x}) \tag{23}$$

We now introduce the shorthand notation

$$\bar{\mathbf{t}}_k = \mu_k(\mathbf{x}) \; and \; \bar{\mathbf{t}} = \sum_{k=1}^{K} \pi_k(\mathbf{x}) \bar{t}_k \tag{24}$$

$$s^2(\mathbf{x}) = E[||\mathbf{t} - E[\mathbf{t}|\mathbf{x}]||^2|\mathbf{x}] \tag{25}$$

$$= \int ||\mathbf{t} - \bar{\mathbf{t}}||^2 p(\mathbf{t}|\mathbf{x})d\mathbf{t} \tag{26}$$

$$= \int (\mathbf{t}^T\mathbf{t} - \mathbf{t}^T\bar{\mathbf{t}} - \bar{\mathbf{t}}^T\mathbf{t} + \bar{\mathbf{t}}^T\bar{\mathbf{t}}) \sum_{k=1}^{K} \pi_k(\mathbf{x}) \mathcal{N}(\mathbf{t}|\mu_k(\mathbf{x}), \sigma_k^2(\mathbf{x}))d\mathbf{t} \tag{27}$$

$$= \sum_{k=1}^{K} \pi_k(\mathbf{x}) \int (\mathbf{t}^T\mathbf{t} - \mathbf{t}^T\bar{\mathbf{t}} - \bar{\mathbf{t}}^T\mathbf{t} + \bar{\mathbf{t}}^T\bar{\mathbf{t}}) \mathcal{N}(\mathbf{t}|\mu_k(\mathbf{x}), \sigma_k^2(\mathbf{x}))d\mathbf{t} \tag{28}$$

Because

$$E[\mathbf{x}\mathbf{x}^T] = \mu\mu^T + \mathbf{\Sigma} \tag{29}$$

So the expression `(20)` can be written as

$$s^2(\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) [\sigma_k^2 + \bar{\mathbf{t}}^T\bar{\mathbf{t}} - \bar{\mathbf{t}}_k^T\bar{\mathbf{t}} - \bar{\mathbf{t}}^T\bar{\mathbf{t}}_k + \bar{\mathbf{t}}_k^T\bar{\mathbf{t}_k}] \tag{30}$$

$$= \sum_{k=1}^{K} \pi_k(\mathbf{x}) [\sigma_k^2 + ||\bar{\mathbf{t}}_k - \bar{\mathbf{t}}||^2] \tag{31}$$

$$= \sum_{k=1}^{K} \pi_k(\mathbf{x}) [\sigma_k^2 + ||\mu_k(\mathbf{x}) - \sum_{l=1}^{K} \pi_l \mu_l(\mathbf{x})||^2] \tag{32}$$

Q4:

The logistic sigmoid function is

$$y(a) = \sigma(a) = \frac{1}{1 + \exp(-a)} \tag{33}$$

Let b be the bias term, w_1 and w_2 be the weights for $\mathbf{A}$ and $\mathbf{B}$, respectively. Then, the logistic threshold unit output y is given by:

$$y = \sigma(w_1\mathbf{A} + w_2\mathbf{B} + b) \tag{34}$$

To match the given Boolean function:

- for the input $(1, 1)$, where $f(\mathbf{A}, \mathbf{B}) = 0$, set $w_1 + w_2 + b < 0$.
- for the input $(0, 0)$, where $f(\mathbf{A}, \mathbf{B}) = 0$, set $b < 0$.
- for the input $(1, 0)$, where $f(\mathbf{A}, \mathbf{B}) = 1$, set $w_1 + b > 0$.
- for the input $(0, 1)$, where $f(\mathbf{A}, \mathbf{B}) = 0$, set $w_2 + b < 0$.

The specific values of the weights can vary as long as they satisfy these conditions. An example is $b = 0.5$, $w_1 = 1$, $w_2 = -1$.

Q5:

(a) number of weights $= 3 * 4 * 4 = 48$.

(b) number of ReLU operations performed on the forward pass $= 3 * 5 * 5 = 75$.

(c) number of weights for the entire network $= 48 + 3 * 5 * 5 * 4 = 348$.

(d) According to Universal Approximation Theorem:

let $\mathbf{C}(\mathbf{X}, \mathbb{R}^m)$ denote the set of continuous functions from s subset $\mathbf{X}$ of a Euclidean $\mathbb{R}^n$. Let $\sigma \in \mathbf{C}(\mathbb{R}, \mathbb{R})$. Note that $\left(\ \sigma \ \circ x \ \right)_i = \sigma(x_i)$, so $\ \sigma \ \circ x$ denotes σ applied to each component of x.

Then σ is not polynomial if and only if for every $n \in \mathbb{N}, m \in \mathbb{N}$, compact $\mathbf{K} \subseteq \mathbb{R}^n, f \in \mathbf{C}(\mathbf{K}, \mathbb{R}^m), \varepsilon > 0$ there exist $k \in \mathbb{N}, \mathbf{A} \in \mathbb{R}^{k \times n}, \mathbf{C} \in \mathbb{R}^{m \times k}$ such that

$$\sup ||f(x) - g(x)|| < \varepsilon \tag{35}$$

where $g(x) = \mathbf{C} \cdot (\ \sigma\ \circ (\mathbf{A} \cdot x + b)\ )$.

In theory, a fully-connected neural network is a universal function approximator.

Universal Approximation Theorem states that a neural network with a single hidden layer can approximate any continuous function, given a sufficiently large number of neurons in that layer. A fully-connected neural network involves multiple hidden layers, each with non-linear activations, enhancing its capability to represent complex classifiers. However, whether it can learn to represent a specific classifier effectively from a given set of training data is another question, which depends on factors like the complexity of the classifier, the quantity and quality of the training data, the design of the network (including the activation functions used, the learning rate, etc.), and the training algorithm used. It is also important to remember that increasing the size or complexity of a network can lead to overfitting if the network is too large relative to the amount of available training data. Overfitting is when the model learns the training data too well, to the point that it performs poorly on new, unseen data. This is why model complexity and size should be carefully managed.

(e) Fully connected neural networks and convolutional neural networks (CNNs) each have their own strengths and weaknesses, and are suited to different types of tasks. Here are some disadvantages of fully connected networks compared to CNNs, especially when dealing with image data:

1. Parameter Efficiency: Fully connected networks do not share parameters across space, leading to a large number of parameters. This can quickly become computationally expensive as the size of the inputs grows. In contrast, CNNs share parameters across space (i.e., the same filter is applied to different parts of the input), significantly reducing the number of parameters.
2. Invariance to Translations: Fully connected networks treat input features independently and don't inherently account for local spatial correlations in the input data. In contrast, CNNs, through the use of convolutional layers, automatically learn and maintain spatial hierarchies, making them better suited to tasks where spatial relationships are important, such as image and video processing.
3. Overfitting: Due to the high number of parameters, fully connected networks are more prone to overfitting, especially when the amount of training data is small. CNNs, by having fewer parameters, can mitigate this issue to some extent.
4. Scalability: Fully connected networks do not scale well to larger images, because the number of parameters increases quadratically with the size of the input. On the other hand, CNNs handle larger images better due to parameter sharing and pooling layers, which reduce the spatial dimensions of the input.

However, it's worth noting that fully connected networks can still perform well on tasks where spatial relationships are not important, or where the input data has been sufficiently preprocessed or engineered. The choice between a fully connected network and a CNN depends on the specific task and the nature of the input data.

Q6:

(a) The output $\mathbf{Y}$ is

$$\begin{aligned}
\mathbf{Y} &= Cw_5(Cw_1\mathbf{X_1} + Cw_3\mathbf{X_2}) + Cw_6(Cw_2\mathbf{X_1} + Cw_4\mathbf{X_2}) && (36)\\
&= C^2(w_1w_5\mathbf{X_1} + w_3w_5\mathbf{X_2} + w_2w_6\mathbf{X_1} + w_4w_6\mathbf{X_2}) && (37)\\
&= C^2(w_1w_5 + w_3w_5)\mathbf{X_1} + C^2(w_2w_6 + w_4w_6)\mathbf{X_2} && (38)\\
&= w_1^{new}\mathbf{X_1} + w_2^{new}\mathbf{X_2} && (39)
\end{aligned}$$

So:

$$\begin{aligned}
w_1^{new} &= C^2(w_1w_5 + w_3w_5)\\
w_2^{new} &= C^2(w_2w_6 + w_4w_6)
\end{aligned} \quad (40)$$

(b) Yes, it is always possible to express a neural network made up of only linear units without a hidden layer, because the composition of linear functions is still a linear function, so any network of linear units can be reduced to a single-layer network.

(c) The output $\mathbf{Y}$ is

$$\begin{aligned}
\mathbf{Y} &= t(w_5\sigma(w_1\mathbf{X_1} + w_3\mathbf{X_2}) + w_6\sigma(w_2\mathbf{X_1} + w_4\mathbf{X_2})) && (41)\\
&= t(\frac{w_5}{1 + \exp(w_1\mathbf{X_1} + w_3\mathbf{X_2})} + \frac{w_6}{1 + \exp(w_2\mathbf{X_1} + w_4\mathbf{X_2})}) && (42)
\end{aligned}$$

The `XOR` of $\mathbf{X_1}$ and $\mathbf{X_2}$ for binary-valued $\mathbf{X_1}$ and $\mathbf{X_2}$ is

$\mathbf{X_1}$	$\mathbf{X_2}$	$\mathbf{Y}$
0	0	0
1	0	1
0	1	1
1	1	0

Let $z_1 = \sigma(w_1\mathbf{X_1} + w_3\mathbf{X_2}), z_2 = \sigma(w_2\mathbf{X_1} + w_4\mathbf{X_2}), z_3 = w_5 z_1 + w_6 z_2$:

$\mathbf{X_1}$	$\mathbf{X_2}$	$\mathbf{Z_1}$	$\mathbf{Z_2}$	$\mathbf{Z_1}$+$\mathbf{Z_2}$	$\mathbf{Y}$
0	0	close to 0	close to 0	close to 0	0
0	1	close to 0	close to 1	close to 1	1
1	0	close to 1	close to 0	close to 1	1
1	1	close to 0	close to 0	close to 0	0

An example is $w_1 = 20, w, w_2 = -10, w_3 = -10, w_4 = 20, w_5 = 10, w_6 = 10$.