Week 4 Discussion 1A

Joe Lin
Learning Assistant

October 25, 2024

Assignment 1 Review

Core Ideas

Problem: Given an image, predict which class it belongs to.

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x?

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N \times D}$

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N\times D}$
- What are the dimensions of y?

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N\times D}$
- What are the dimensions of y? $\mathbb{R}^{N\times 1}$

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N \times D}$
- What are the dimensions of y? $\mathbb{R}^{N \times 1}$
- What are the dimensions of W?

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N \times D}$
- What are the dimensions of y? $\mathbb{R}^{N \times 1}$
- What are the dimensions of W? $\mathbb{R}^{1 \times D}$

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N \times D}$
- What are the dimensions of y? $\mathbb{R}^{N \times 1}$
- What are the dimensions of W? $\mathbb{R}^{1 \times D}$

Weaknesses?

Core Ideas

Problem: Given an image, predict which class it belongs to.

Solution 1: **Linear Regression**, which aims to fit the observed data (x, y) with a linear model $\rightarrow y = xW^T + b$.

Dimension Check

- What are the dimensions of x? $\mathbb{R}^{N\times D}$
- What are the dimensions of y? $\mathbb{R}^{N\times 1}$
- What are the dimensions of W? $\mathbb{R}^{1\times D}$

Weaknesses? Discrete classes, but we predict all real values.

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function.

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Dimension Check

- What are the dimensions of y?

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Dimension Check

- What are the dimensions of y? $\mathbb{R}^{N \times C}$

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Dimension Check

- What are the dimensions of y? $\mathbb{R}^{N\times C}$
- What are the dimensions of W?

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Dimension Check

- What are the dimensions of y? $\mathbb{R}^{N \times C}$
- What are the dimensions of W? $\mathbb{R}^{C \times D}$

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \to y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Dimension Check

- What are the dimensions of y? $\mathbb{R}^{N \times C}$
- What are the dimensions of W? $\mathbb{R}^{C \times D}$

Weaknesses?

Core Ideas

Solution 2: **Logistic Regression** uses logistic (sigmoid) function to fit observed data $(x, y) \rightarrow y = \sigma(xW^T + b)$.

Recall the formula for logistic function. $\sigma(z) = \frac{1}{1+e^{-z}}$

Dimension Check

- What are the dimensions of y? $\mathbb{R}^{N \times C}$
- What are the dimensions of W? $\mathbb{R}^{C \times D}$

Weaknesses? Predictions do not form a probability distribution over the classes.

Core Ideas

Solution 3: **Softmax Regression** uses softmax function to fit observed data $(x, y) \to y = \mathrm{softmax}(xW^T + b)$.

Recall the formula for softmax function.

Core Ideas

Solution 3: **Softmax Regression** uses softmax function to fit observed data $(x, y) \rightarrow y = \text{softmax}(xW^T + b)$.

Recall the formula for softmax function. $\text{softmax}(x) = \frac{e^z}{\sum_{j=1}^{C} e^{z_j}}$

Core Ideas

Solution 3: **Softmax Regression** uses softmax function to fit observed data $(x, y) \to y = \mathrm{softmax}(xW^T + b)$.

Recall the formula for softmax function. $\mathrm{softmax}(x) = \frac{e^z}{\sum_{j=1}^{C} e^{z_j}}$

Dimension Check

- What are the dimensions of y?

Core Ideas

Solution 3: **Softmax Regression** uses softmax function to fit observed data $(x, y) \to y = \mathrm{softmax}(xW^T + b)$.

Recall the formula for softmax function. $\mathrm{softmax}(x) = \frac{e^z}{\sum_{j=1}^{C} e^{z_j}}$

Dimension Check

- What are the dimensions of y? $\mathbb{R}^{N \times C}$

Implementation

```
self.W = torch.zeros(..., requires_grad=True).to(device)
```

Implementation

```
self.W = torch.zeros(..., requires_grad=True).to(device)
```

What happens when we try to backpropagate and compute gradients with respect to W ($\frac{\partial \mathcal{L}}{\partial W}$)?

Implementation

```
self.W = torch.zeros(..., requires_grad=True).to(device)
```

What happens when we try to backpropagate and compute gradients with respect to W ($\frac{\partial \mathcal{L}}{\partial W}$)? Unable to access gradients because `self.W` is not a leaf tensor

How do we fix this?

Implementation

```
self.W = torch.zeros(..., requires_grad=True).to(device)
```

What happens when we try to backpropagate and compute gradients with respect to W ($\frac{\partial \mathcal{L}}{\partial W}$)? Unable to access gradients because `self.W` is not a leaf tensor

How do we fix this?

```
self.W = torch.zeros(..., requires_grad=True, device=device)
```

Implementation

```
cross_entropy_loss = -torch.sum(y * torch.log(p), dim=-1)
```

Implementation

```
cross_entropy_loss = -torch.sum(y * torch.log(p), dim=-1)
```

Recall the cross entropy formula. $-\log p_y$, where p_y is the predicted probability that image belongs to the ground truth class

Implementation

```
cross_entropy_loss = -torch.sum(y * torch.log(p), dim=-1)
```

Recall the cross entropy formula. $-\log p_y$, where p_y is the predicted probability that image belongs to the ground truth class

$$y = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$p = \begin{pmatrix} 0.2 & 0.1 & 0.4 & 0.2 & 0.1 \end{pmatrix}$$

$$\mathcal{L}_{\text{ce}} = -(0 \cdot \log 0.2 + 0 \cdot \log 0.1 + 1 \cdot \log 0.4 + 0 \cdot \log 0.2 + 0 \cdot \log 0.1)$$

What issues may arise with this implementation?

Implementation

At start of training, what would p likely be?

Implementation

At start of training, what would p likely be?

$p = (0.2, 0.2, 0.2, 0.2, 0.2)$

Implementation

At start of training, what would p likely be?

$p = (0.2, 0.2, 0.2, 0.2, 0.2)$

How about after training for a while?

Implementation

At start of training, what would p likely be?

$p = (0.2, 0.2, 0.2, 0.2, 0.2)$

How about after training for a while?

$p = (0.001, 0, 0.99, 0.005, 0.004)$

Implementation

At start of training, what would p likely be?

$p = (0.2, 0.2, 0.2, 0.2, 0.2)$

How about after training for a while?

$p = (0.001, 0, 0.99, 0.005, 0.004)$

Why is this troublesome?

How can we fix this?

Implementation

At start of training, what would p likely be?

$p = (0.2, 0.2, 0.2, 0.2, 0.2)$

How about after training for a while?

$p = (0.001, 0, 0.99, 0.005, 0.004)$

Why is this troublesome? $\log 0$ is undefined

How can we fix this?

Implementation

At start of training, what would p likely be?

$p = (0.2, 0.2, 0.2, 0.2, 0.2)$

How about after training for a while?

$p = (0.001, 0, 0.99, 0.005, 0.004)$

Why is this troublesome? $\log 0$ is undefined

How can we fix this?

```
cross_entropy_loss = -torch.log(torch.sum(y * p, dim=-1))
```

Assignment 2 Preview

Convolutions

What are some configurable hyperparameters of convolutions?

Convolutions

What are some configurable hyperparameters of convolutions?

Kernel size, padding, stride, ...

Why use convolutions?

Convolutions

What are some configurable hyperparameters of convolutions?

Kernel size, padding, stride, ...

Why use convolutions? Shared parameters, computational efficiency, takes advantage of inherent structure of data (for smaller datasets and model size)

Let's practice.