

Where we are...

$$s = f(x; W) = Wx$$

scores function

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

SVM loss

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k W_k^2$$

data loss + regularization

want $\nabla_W L$

we have this core function we looked at several Los

Where we are...

$$s = f(x; W) = Wx$$

scores function

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

SVM loss

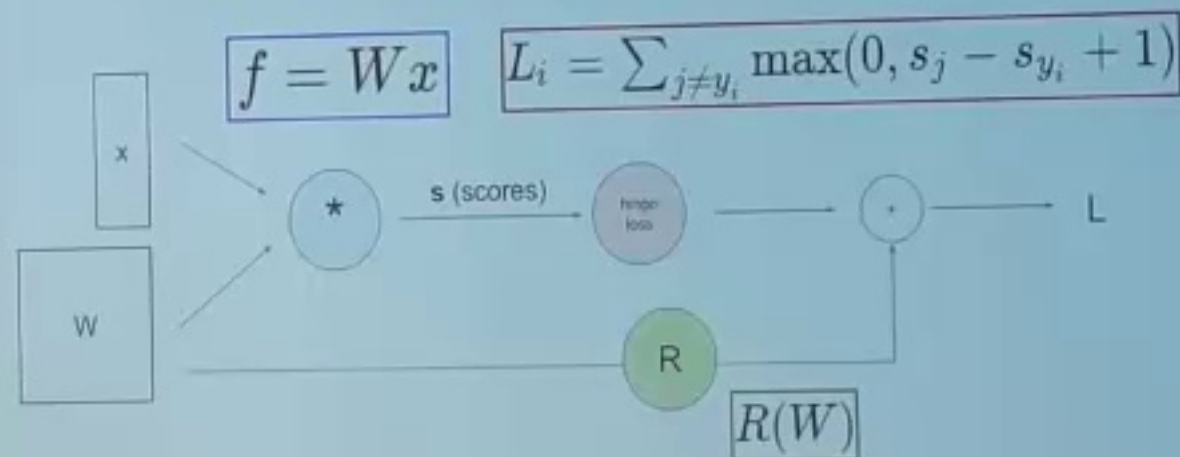
$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k W_k^2$$

data loss + regularization

want $\nabla_W L$

functions such as the SB must function last time and

Computational Graph



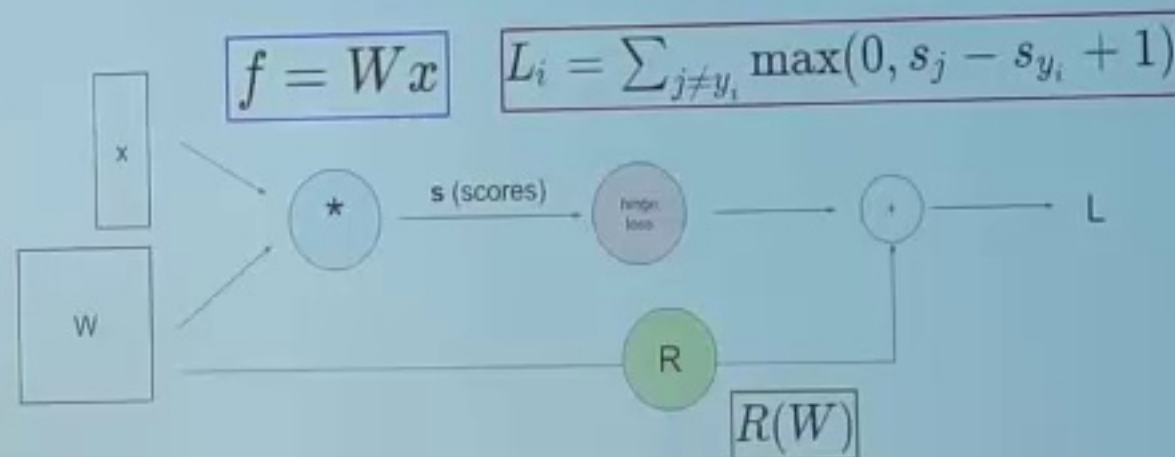
Fei-Fei Li & Andrej Karpathy & Justin Johnson

Lecture 4 - 6

13 Jan 2016

where we are right evaluating on your weights doing

Computational Graph



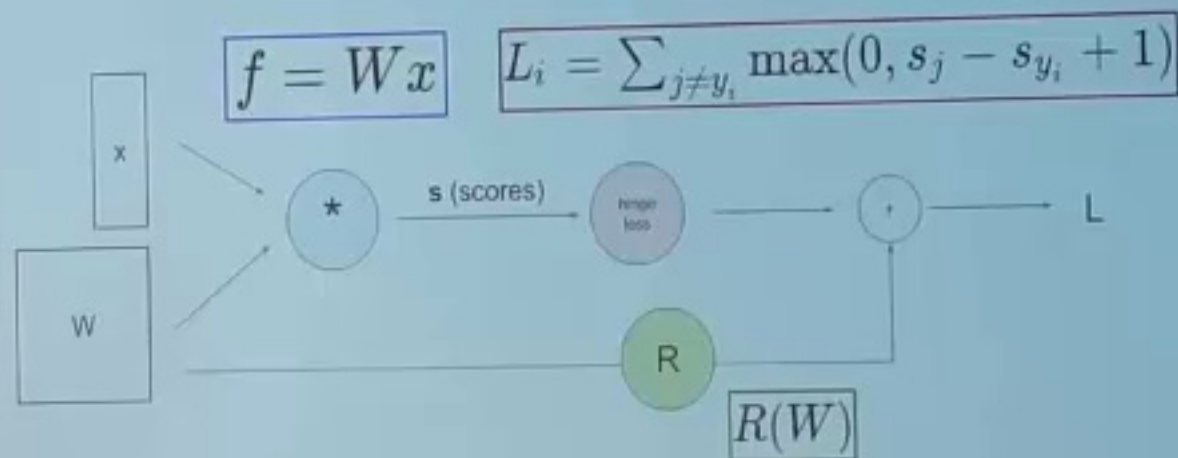
Fei-Fei Li & Andrej Karpathy & Justin Johnson

Lecture 4 - 6

13 Jan 2016

primer update and is repeating this over and over

Computational Graph



Fei-Fei Li & Andrej Karpathy & Justin Johnson

Lecture 4 - 6

13 Jan 2016

again so that we're converging to the low points of

Where we are...

$$s = f(x; W) = Wx$$

scores function

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

SVM loss

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k W_k^2$$

data loss + regularization

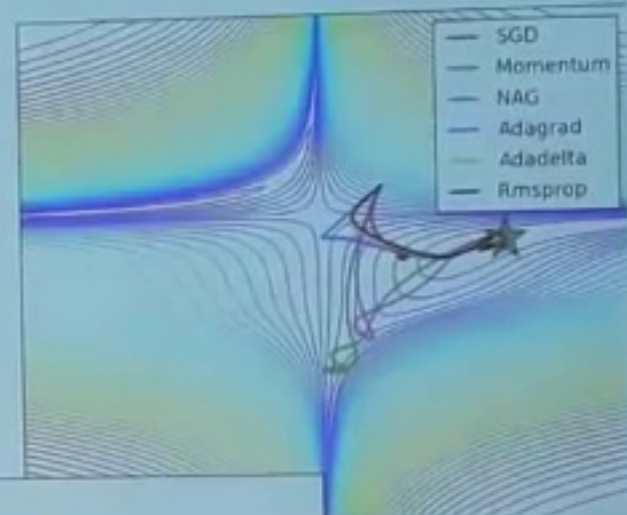
want $\nabla_W L$

we look at the full loss that you achieve for any

Optimization



```
while True:
    weights_grad = evaluate_gradient(loss_fun, data, weights)
    weights += - step_size * weights_grad
```



(image credits
to Alec Radford)

Fei-Fei Li & Andrej Karpathy & Justin Johnson

Lecture 4 - 4

13 Jan 2016

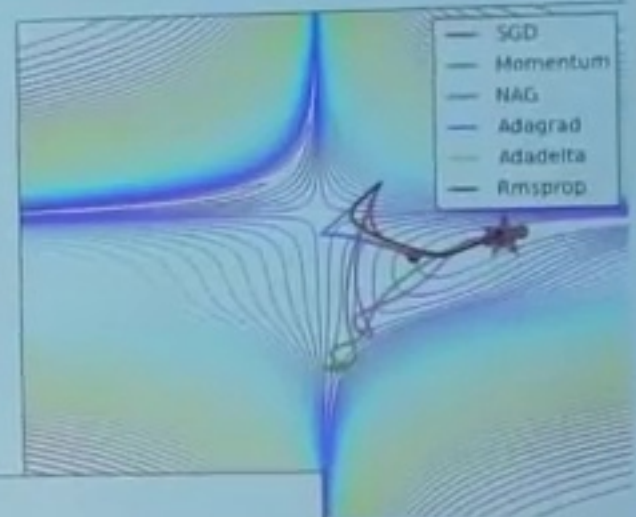
particular set of weights over your training data and

Optimization



• David Lae, Johannes Becher

```
while True:
    weights_grad = evaluate_gradient(loss_fun, data, weights)
    weights += - step_size * weights_grad
```



(image credits
to Alec Radford)

Fei-Fei Li & Andrej Karpathy & Justin Johnson

Lecture 4 - 4

13 Jan 2016

this last minute to compose in regulation loss and

Gradient Descent

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Numerical gradient: slow :(, approximate :(, easy to write :)

Analytic gradient: fast :), exact :), error-prone :(

In practice: Derive analytic gradient, check your implementation with numerical gradient

really what we want to do is we wanted to write not

Gradient Descent

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Numerical gradient: slow :(, approximate :(, easy to write :)

Analytic gradient: fast :), exact :), error-prone :(

In practice: Derive analytic gradient, check your implementation with numerical gradient

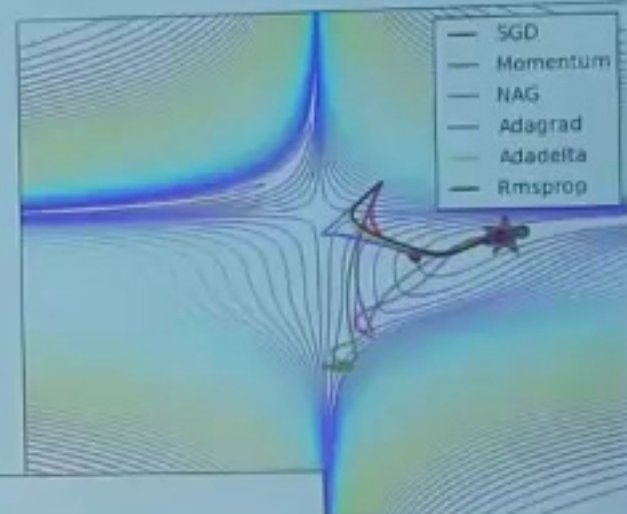
be greedy and expression of the last function with

Optimization



• negative gradient descent

```
while True:
    weights_grad = evaluate_gradients(loss_func, data, weights)
    weights -= step_size * weights_grad
```



(image credits
to Alec Radford)

Fei-Fei Li & Andrej Karpathy & Justin Johnson

Lecture 4 - 4

13 Jan 2016

respect to the weights and we want to do this so that

Where we are...

$$s = f(x; W) = Wx$$

scores function

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

SVM loss

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k W_k^2$$

data loss + regularization

want $\nabla_W L$

we can actually perform the optimization process

Gradient Descent

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Numerical gradient: slow :(, approximate :(, easy to write :)

Analytic gradient: fast :), exact :), error-prone :(

In practice: Derive analytic gradient, check your implementation with numerical gradient

optimization process we're doing reading this and