Components of CNNs

MPCS 53111

May 2017

1 Symbolic differentiation

Suppose we have a list of objects S, each object of S has two methods forward() and backward(), and S is always kept in the topological order of the computation graph of our neural networks. S has 3 operations:

function Add(S, Object o):
01. $S \leftarrow S \cup \{o\}$
end function

function Forward(S):
01. for each $o \in S$:
02. o.forward()
03. end for
end function

function Backward(S):
01. for each $o \in S$ in the reversed order:
02. o.backward()
03. end for
end function

The generic of constructing a neural network, training and testing can described in a high-level language as follows:

procedure Neural Network Scheme():
01. **Construction**
02. Initialize the computation graph as empty $S \leftarrow \emptyset$
03. Add all the layers we want to S in the bottom-up order
04. **Training**
05. for each training example x (or a batch of training examples):
06. S.forward()
07. S.backward()
08. Gradient Descent Algorithm
09. end for

10. **Testing**
11. for each testing example x:
12. S.forward()
13. Return the prediction for example x
14. end for

end procedure

2 Convolution Forward and Backward

Read the another document that published before!

3 Padding Forward and Backward

Padding Forward: Read the another document that published before!

Padding Backward:

function Backward($\mathcal{L}'.\delta \in \Re^{B\times H'\times W'\times D}$, $w \in \mathbb{N}_{odd}$, $h \in \mathbb{N}_{odd}$)
01. $\mathcal{L}.\delta[:,:,:,:] \leftarrow \mathcal{L}.\delta[:,:,:,:] + \mathcal{L}'.\delta[:, \lfloor h/2 \rfloor : H + \lfloor h/2 \rfloor, \lfloor w/2 \rfloor : W + \lfloor w/2 \rfloor, :]$

end function

4 Max-Pooling Forward and Backward

Max-Pooling Forward: Read the another document that published before!

Max-Pooling Backward:

function Backward($\mathcal{L}'.\delta \in \Re^{B\times H'\times W'\times D}, w \in \mathbb{N}, h \in \mathbb{N}$)
01. for each $x = 0 \rightarrow H' - 1$:
02. for each $y = 0 \rightarrow W' - 1$:
03. $(u, v) \leftarrow$ **max-index** $(\mathcal{L}[:, x * h : (x + 1) * h, y * w, (y + 1) * w, :])$
06. $\mathcal{L}.\delta[:, u, v, :] \leftarrow \mathcal{L}.\delta[:, u, v, :] + \mathcal{L}'.\delta[:, x, y, :]$
07. end for
08. end for

end function

5 Average-Pooling Forward and Backward

Average-Pooling Forward: Read the another document that published before!

Average-Pooling Backward:

function Average-Pooling($\mathcal{L}'.\delta \in \Re^{B \times H' \times W' \times D}, w \in \mathbb{N}, h \in \mathbb{N}$)
01. for each $x = 0 \rightarrow H' - 1$:
02. for each $y = 0 \rightarrow W' - 1$:
03. $\mathcal{X} \leftarrow x * h : (x + 1) * h$
04. $\mathcal{Y} \leftarrow y * w : (y + 1) * w$
05. $\mathcal{L}.\delta[:, \mathcal{X}, \mathcal{Y}, :] \leftarrow \mathcal{L}.\delta[:, \mathcal{X}, \mathcal{Y}, :] + \frac{1}{w \cdot h}\mathcal{L}'.\delta[:, x, y, :]$
06. end for
07. end for
end function

6 Hinge Loss

function Forward(Target y, Prediction $\hat{y}$)
01. **return** $max(0, 1 - y \cdot \hat{y})$
end function

function Backward(Target y, Prediction $\hat{y}$)
01. if $y \cdot \hat{y} > 1$:
02. $\hat{y}.\delta \leftarrow 0$
03. else:
04. $\hat{y}.\delta \leftarrow -y$
05. end if
end function