U-net CNN in APL: Appendices

Exploring zero-framework, zero-library machine learning

ANONYMOUS AUTHOR(S)

ACM Reference Format:
Anonymous Author(s). 2022. U-net CNN in APL: Appendices: Exploring zero-framework, zero-library machine learning. In . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ICFP'22, Sep 11 - Sep 16, 2022, Ljubljana, Slovenia

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

APPENDIX A: COMPLETE APL U-NET IMPLEMENTATION

```
:Namespace UNET

 W←⍬ ⋄ V←⍬ ⋄ Z←⍬ ⋄ LR←1e¯9 ⋄ MO←0.99

 FWD←{Z⊢←(≢W)⍴⊂⍬
  CV←{0⌈z⊣Z[α]←⊂Z[α],⊂z←(,[2+⍳3]3 3⌺⊃Z[α]←⊂ω)+.×,[⍳3]α⊃W}
  CC←{ω,⍨(⌊p)↓(-⌈p)↓(α⊃Z)⊣p←2÷⍨(ρα⊃Z)-ρω}
  MX←{⌈⌿[2],[2 3](2 2ρ2)⌺⊃Z[α]←⊂ω}
  UP←{((2×¯1↓ρω),¯1↑ρα⊃W)ρ0 2 1 3 4⍉ω+.×α⊃W⊣Z[α]←⊂ω}
  C1←{1E¯8+z÷[⍳2]+/z←*z-[⍳2]⌈/z←ω+.×α⊃W⊣Z[α]←⊂ω}
  LA←{α≥≢Z:ω
    down←(α+6)∇(α+2)MX(α+1)CV(α+0)CV ω
    (α+2)CC(α+5)UP(α+4)CV(α+3)CV down}
  2 C1 1 CV 0 CV 3 LA ωρ⍨3↑1,⍨ρω}

 BCK←{Y←α ⋄ YΔ←ω
  Δ←{0⊣W[α]←⊂(α⊃W)-LR×⊃V[α]←⊂ω+MO×(ρω)ρα⊃V}
  ΔCV←{w←,[⍳3]⊖⌽[1]0 1 3 2⍉α⊃W ⋄ x←⊃α⊃Z ⋄ Δz←ω×0<1⊃α⊃Z
   ΔZ←¯2⊖¯2⌽[1](4+2↑ρΔz)↑Δz
   _←α Δ 3 0 1 2⍉(⍉,[⍳2]Δz)+.×,[⍳2]3 3⌺x
   w+.×⍨,[2+⍳3]3 3⌺ΔZ}
  ΔCC←{x←α⊃Z ⋄ Δz←ω ⋄ d←-⌊2÷⍨2↑(ρx)-ρΔz ⋄ (⊃d)⊖(1⊃d)⌽[1](ρx)↑Δz}
  ΔMX←{x←α⊃Z ⋄ Δz←ω ⋄ y×x=y←(ρx)↑2⌿2/[1]Δz}
  ΔUP←{w←α⊃W ⋄ x←α⊃Z ⋄ Δz←ω ⋄ cz←(2 2ρ2)⌺Δz
   _←α Δ(⍉,[⍳2]x)+.×,[⍳2]cz
   (,[2+⍳3]cz)+.×⍉⍪w}
  ΔC1←{w←α⊃W ⋄ x←α⊃Z ⋄ Δz←ω ⋄ _←α Δ(⍉,[⍳2]x)+.×,[⍳2]Δz ⋄ Δz+.×⍉w}
  ΔLA←{α≥≢Z:ω
   down←(α+6)∇(α+3)ΔCV(α+4)ΔCV(α+5)ΔUP ω↑[2]⍨-2÷⍨⊃⌽ρω
   (α+0)ΔCV(α+1)ΔCV(ω ΔCC⍨α+2)+(α+2)ΔMX down}
  3 ΔLA 0 ΔCV 1 ΔCV 2 ΔC1 YΔ-(~Y),[1.5]Y}

 E←{-+⌿,⍟(α×ω[;;1])+(~α)×ω[;;0]}

 RUN←{Y YΔ(Y E YΔ)⊣(Y←⌊0.5+nm↑ω↓⍨2÷⍨(ρω)-nm←2↑ρYΔ)BCK⊢YΔ←FWD α}

:EndNamespace
```

APPENDIX B: PYTORCH REFERENCE IMPLEMENTATION

```python
import torch
import torch.nn as nn
import torchvision
import torchvision.transforms.functional

class TwoConv(nn.Module):

   def __init__(self, in_channels, out_channels):
        super().__init__()

        self.path = nn.Sequential(
            nn.Conv2d(in_channels, out_channels,
                kernel_size=(3, 3), bias=False),
            nn.ReLU(inplace=True),
            nn.Conv2d(out_channels, out_channels,
                kernel_size=(3, 3), bias=False),
            nn.ReLU(inplace=True),
        )

    def forward(self, x):
        return self.path(x)

class Down(nn.Module):

    def __init__(self, in_channels):
        super().__init__()

        self.path = nn.Sequential(
            nn.MaxPool2d(kernel_size=(2, 2), stride=2),
            TwoConv(in_channels, 2 * in_channels),
        )

    def forward(self, x):
        return self.path(x)

class Up(nn.Module):

    def __init__(self, in_channels):
        super().__init__()

        self.upsampling = nn.ConvTranspose2d(
            in_channels,
            in_channels // 2,
            kernel_size=(2, 2),
            stride=2,
            bias=False,
```

```
        )
        self.convolutions =
            TwoConv(in_channels, in_channels // 2)

    def forward(self, x_to_crop, x_in):

        upped = self.upsampling(x_in)
        cropped = torchvision.transforms.functional.center_crop(
            x_to_crop, upped.shape[-2:]
        )
        x = torch.cat([cropped, upped], dim=1)
        return self.convolutions(x)

class USegment(nn.Module):

    def __init__(self, in_channels, bottom_u=None):
        super().__init__()

        # Default value for the bottom U.
        if bottom_u is None:
            bottom_u = lambda x: x

        self.down = Down(in_channels)
        self.bottom_u = bottom_u
        self.up = Up(2 * in_channels)

    def forward(self, x):
        return self.up(x, self.bottom_u(self.down(x)))

class UNet(nn.Module):

    def __init__(self):
        super().__init__()

        self.u = USegment(512)
        self.u = USegment(256, self.u)
        self.u = USegment(128, self.u)
        self.u = USegment(64, self.u)
        self.path = nn.Sequential(
            TwoConv(1, 64),
            self.u,
            nn.Conv2d(64, 2, kernel_size=1, bias=False),
        )

    def forward(self, x):
        return self.path(x)
```