

$$(X, O) = e^{-\frac{x^2}{2\sigma^2}}$$

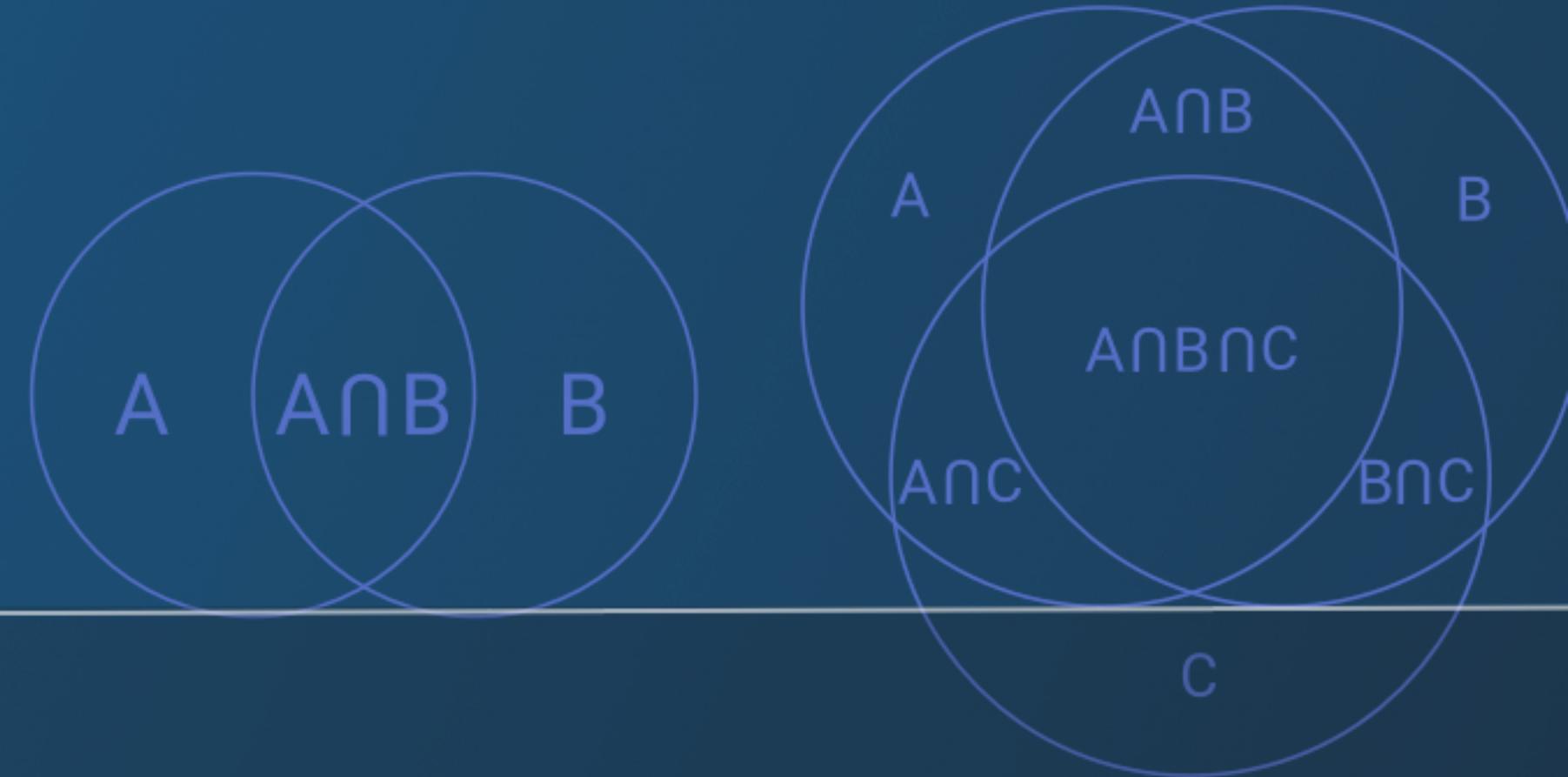
$$x(X, O) = -\frac{x}{\sigma^2} G(X, O) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

$$xx(X, O) = \frac{x^2 - \sigma^2}{\sigma^4} G(X, O) = \frac{x^2 - \sigma^2}{\sigma^4} e^{-\frac{x^2}{2\sigma^2}}$$

$$xxx(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} G(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} e^{-\frac{x^2}{2\sigma^2}}$$

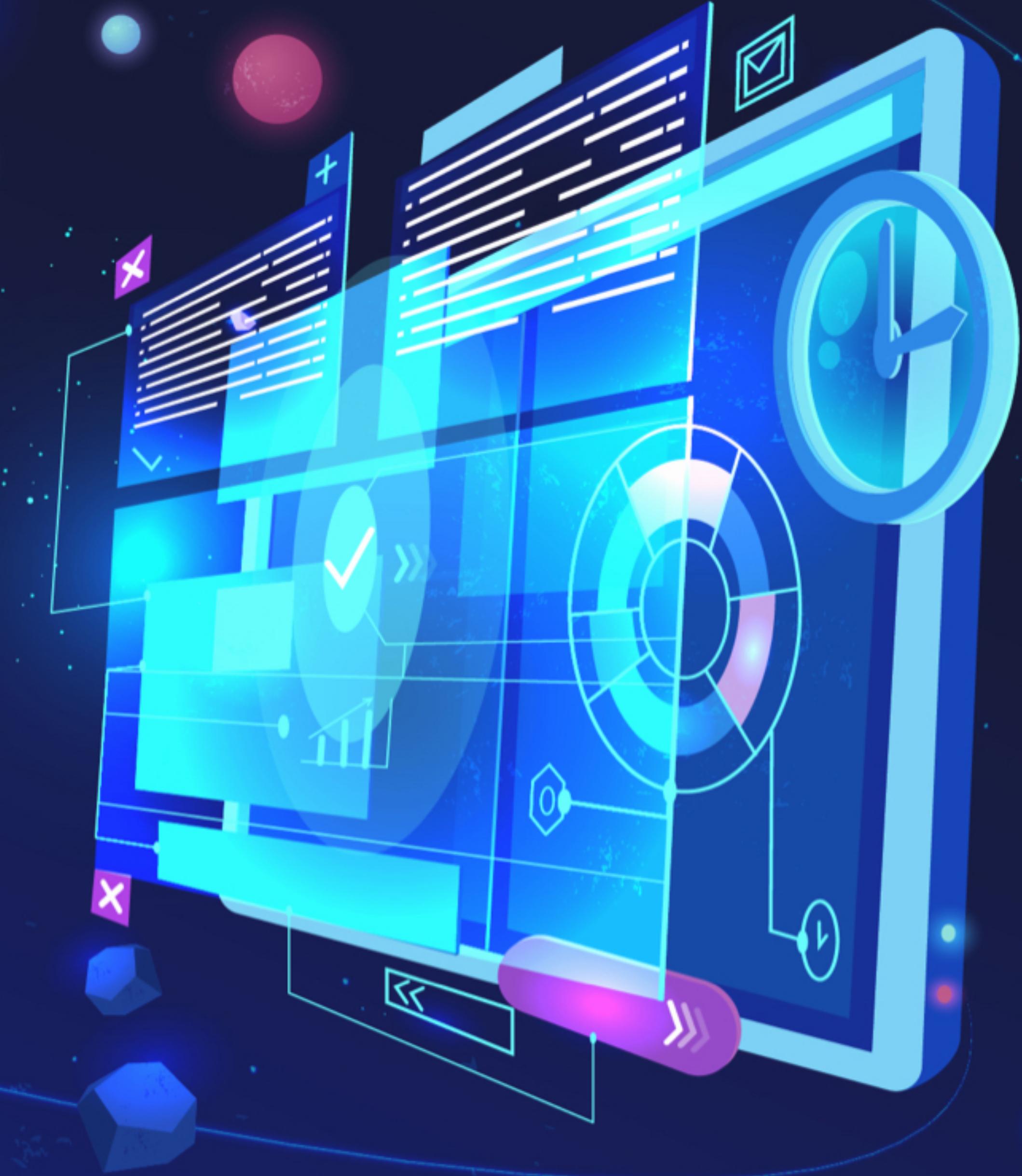
# Julia 程式語言學習馬拉松

Day 33



cupay 陪跑專家 : Andy Tu

# 卷積神經網路模型簡介





# 重要知識點



- 了解如何使用多樣的網路層
- 了解如何搭建卷積神經網路
- 了解如何使用預訓練模型



# 卷積層

- 在 Flux 中使用卷積層非常簡單。
- 創建 Conv 物件，第一個參數為 kernel 的大小，第二個參數為輸入到輸出的維度，第三個參數為 activation function

```
Conv((2, 2), 1=>16, relu)
```



# Maxpooling

- 創建 maxpooling layer，MaxPool 的第一個參數為 kernel 的大小，並且支援 pad 及 stride 參數。

```
MaxPool((2,2))  
MaxPool((3,3), pad=(0,0),  
        stride=(2,2))
```



# 卷積神經網路



```
model = Chain(  
    Conv((3, 3), 1=>16, pad=(1,1), relu),  
    MaxPool((2,2)),  
    Conv((3, 3), 16=>32, pad=(1,1), relu),  
    MaxPool((2,2)),  
    Conv((3, 3), 32=>32, pad=(1,1), relu),  
    MaxPool((2,2)),  
    x -> reshape(x, :, size(x, 4)),  
    Dense(288, 10),  
    softmax)
```



# 使用 CUDA



- Flux 對於 CUDA 的支援來自於 CuArrays 套件，只要載入 CuArrays 套件便自動啟用 CUDA 的支援。
- 唯一使用者需要做的事情就是把模型與資料傳送到 CUDA 上。

```
data = data |> gpu  
model = model |> gpu
```



# 訓練卷積神經網路



- 卷積神經網路的訓練與多層感知器的訓練相同。

```
loss(x, y) = crossentropy(model(x),  
y)  
dataset = repeated((X, Y), 200)  
evalcb() = @show loss(X, Y)  
Flux.train!(loss, params(model),  
dataset, ADAM(), cb=throttle(evalcb,  
10))
```



# 模型評估

- 模型評估方式與多層感知器的評估相同。

```
using Flux: onecold
accuracy(x, y) = mean(onecold(m(x)) .==
onecold(y))
accuracy(X, Y)
```



# Batch normalization

- Batch normalization 的使用也是如同網路層一樣疊上去。
- BatchNorm 的第一個參數對應前一層的輸出維度，第二個參數可以允許使用 activation function。

```
BatchNorm(64, relu)
```



# Dropout



- Dropout 的使用如同網路層一樣疊上去。
- Dropout 需要設定神經元被隱藏的機率  $p$ 。
- Dropout 的使用主要目的在於避免 overfitting。

Dropout( $p$ )



# 使用預訓練模型



- 在 Metalhead.jl 套件中包含知名的預訓練模型，安裝套件之後即可使用。

```
using Metalhead
using Metalhead: classify

vgg = VGG19()
classify(vgg, img)
```

# 知識點 回顧

- 了解如何使用多樣的網路層
- 了解如何搭建卷積神經網路
- 了解如何使用預訓練模型



## 推薦閱讀

- [Flux.jl 官方網頁](#)
- [Metalhead.jl Github](#)





解題時間

請跳出 PDF 至官網 Sample Code  
& 作業開始解題