Good to know

1. Where and why ReLU outperforms Sigmoid in hidden layers

1. **Alleviation of the vanishing-gradient problem**
   - **Sigmoid**:

     $$\sigma(x) = \frac{1}{1 + e^{-x}}, \quad \sigma'(x) = \sigma(x)\,\big(1 - \sigma(x)\big) \leq 0.25.$$

     For large $|x|$, $\sigma'(x) \to 0$, so gradients vanish as they backpropagate through many layers.
   - **ReLU**:

     $$\mathrm{ReLU}(x) = \max(0,\, x), \quad \mathrm{ReLU}'(x) = \begin{cases} 1, & x > 0, \\ 0, & x \leq 0. \end{cases}$$

     When active ($x > 0$), the gradient is constant 1, so it preserves gradient magnitude better in deep nets.
2. **Sparse activations and computational efficiency**
   - ReLU sets all negative pre-activations to zero, producing sparsity in the network. Sparse activations reduce inter-neuron dependencies and can speed up both forward and backward passes.
   - The piecewise-linear form of ReLU is cheaper to compute than the exponential in $\sigma(x)$.
3. **Empirical performance in deep architectures**
   - Modern deep convolutional and feed-forward networks (e.g. AlexNet, ResNet) almost universally use ReLU (or its variants), because it leads to faster convergence and often better generalization.

---

2. When Logistic (Sigmoid) is preferable

1. **Output layers for probabilistic interpretation**
   - For binary classification, the output neuron often needs to produce a probability in $[0, 1]$. The sigmoid's range matches this requirement directly, making it the natural choice for the final layer when used with cross-entropy loss.
2. **Gating mechanisms in recurrent networks**
   - In architectures like LSTMs or GRUs, gates (forget, input, output) must produce values in $[0, 1]$ to modulate information flow. Logistic activations perfectly serve this gating role, whereas ReLU's unbounded positive output and zero gradient for negatives would break the soft gating behavior.
3. **Shallow or specialized models**
   - In some shallow networks or autoencoders where depth is limited, vanishing gradients are less severe. Sigmoids can still work reasonably, especially when you need bounded, smooth outputs or want to exploit their probabilistic interpretation in energy-based models (e.g. Boltzmann machines).

Mathematical summary

Property	Sigmoid $\sigma(x)$	ReLU $\max(0, x)$
Range	$(0, 1)$	$[0, \infty)$
Derivative	$\sigma(x)(1 - \sigma(x)) \leq 0.25$	1 for $x > 0$, else 0
Vanishing gradient	Severe for (	x
Sparsity	No	Yes (zeros for $x \leq 0$)
Computational cost	Higher (exponential)	Lower (simple comparison)

By choosing ReLU in deep hidden layers, you maintain stronger gradients, encourage sparsity, and speed up training. Conversely, Sigmoid remains indispensable where bounded outputs or soft gating are required.