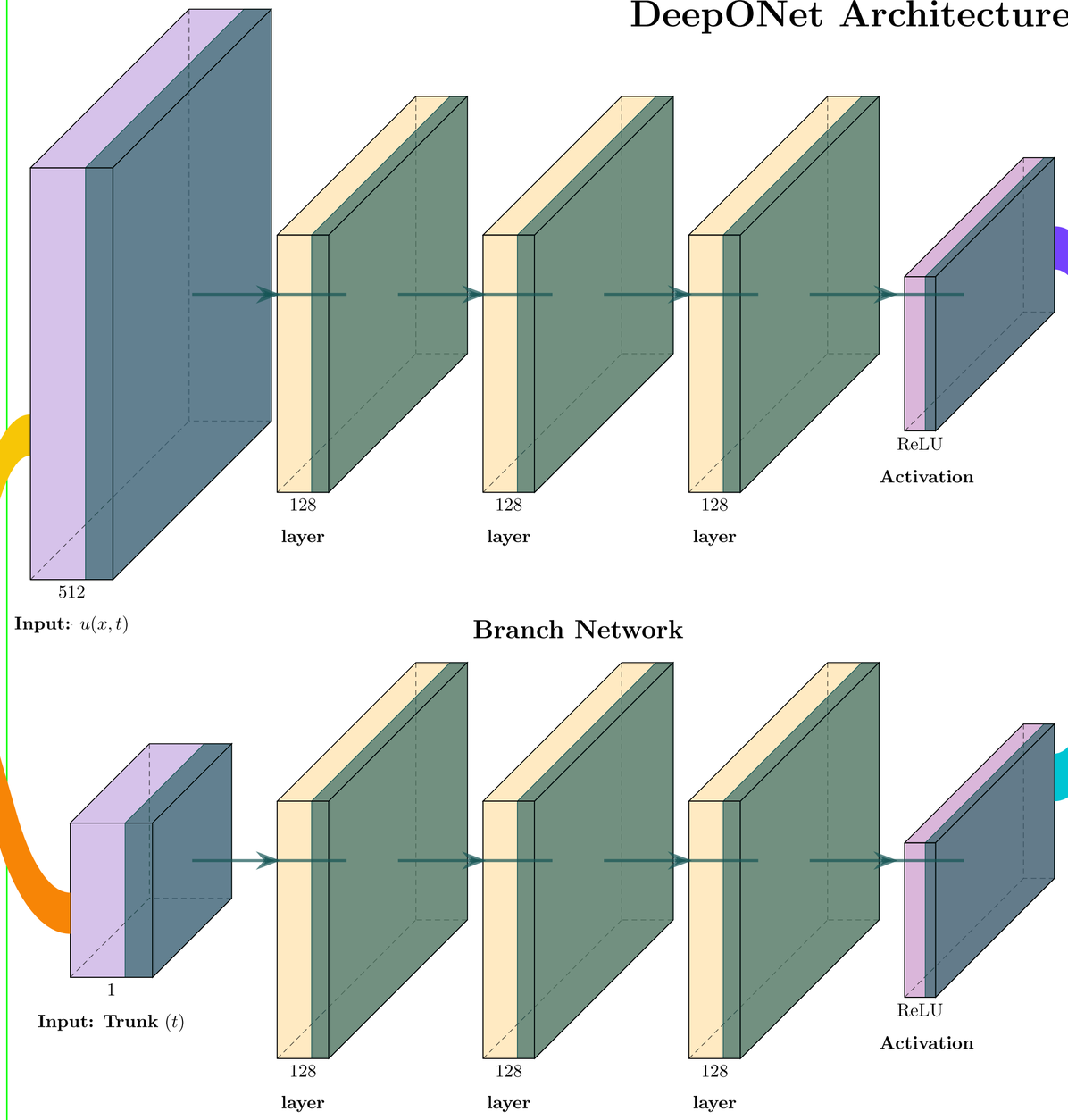
Data Generation (KS Simulator)

For each sample:
- $u(x,t) \leftarrow$ initial condition + noise
- $t \leftarrow$ random time in $[0, T]$
- $\nabla u \leftarrow$ gradient of $u(x,t)$ (via FFT)
- $f(x,t) = -u(x,t) - 0.1\nabla u(x,t)$

(target control signal)

DeepONet Architecture



Element-wise Multiply:
$z = \text{Branch}(u) \odot \text{Trunk}(t)$

Element-wise Multiply:
$z = \text{Branch}(u) \odot \text{Trunk}(t)$

Training Loss:
Loss: MSE(predicted $f(x,t)$, target $f(x,t)$)
Optimizer: Adam, lr = 0.001
3000 epochs, batch updates

Trained DeepONet:
Ready for use in RL

State: $u(x,t) \in \mathbb{R}^{512}$

Actor Network (DDPG Policy):
- **Input:** $u(x,t)$
- **Output:** $a(x,t) \in [-1,1]^{512}$

KS Equation Solver (Spectral):
- Applies $c(x,t)$ to evolve u

Compute Reward r:
Based on:
- Energy
- Action magnitude
- $\partial u/\partial x$ and $\partial u/\partial t$ penalties

Critic Network (Q-function):
Evaluates state-action pair, $Q(s_t, a_t)$

Experience stored in Replay Buffer

Policy and Critic updated via gradients