EDteam

Frameworks y Librerías para Redes Neuronales

¿Cómo se desarrollan algoritmos de AI?

Muchas herramientas que utilizamos se han creado sobre otras herramientas

Se refiere a la arquitectura de la red neuronal, no a los datos usados

EDteam

A mayor **Flexibilidad**, menor **Simplicidad**

A mayor **Simplicidad**, menor **Flexibilidad**

Desde 0:
Solo Python

Se deberían desarrollar todas las funciones y módulos de la red, desde 0

```python
def initialize_parameters():
    """Initialize network parameters (weights and biases)."""
new *
def forward_propagation():
    """Perform forward propagation to compute activations."""
new *
def compute_cost():
    """Compute the cost (loss) after forward propagation."""
new *
def backward_propagation():
    """Compute gradients for backpropagation."""
new *
def update_parameters():
    """Update network parameters using computed gradients."""
new *
def train():
    """Train the neural network over multiple iterations."""
new *
def predict():
    """Make predictions using the trained model."""
new *
def activation_function():
    """Implement the activation function (e.g., ReLU, sigmoid, tanh)."""
new *
def activation_derivative():
    """Compute the derivative of the activation function for backpropagation."""
new *
def initialize_velocity():
    """Initialize velocity vectors for optimization with momentum."""
new *
def initialize_adam():
    """Initialize parameters for the Adam optimization algorithm."""
new *
def optimize():
    """Execute the optimization process (e.g., SGD, momentum, Adam)."""
new *
def plot_cost():
    """Plot the cost function over time to visualize training progress."""
new *
def shuffle_data():
    """Shuffle the dataset before each epoch."""
new *
def split_data():
    """Split the dataset into training, validation, and test sets."""
new *
def save_model():
    """Save the trained model parameters to a file."""
new *
def load_model():
    """Load the model parameters from a saved file."""
```

...

EDteam

Pros

Mayor **control**

Mayor **aprendizaje**

Contras

Difícil de **mantener** a lo largo del tiempo

Reinventar la rueda

Es probable que no esté bien **optimizado**

Muy acoplada a mi problema, poco **adaptable**

Mal en relación **costo / beneficio**

EDteam

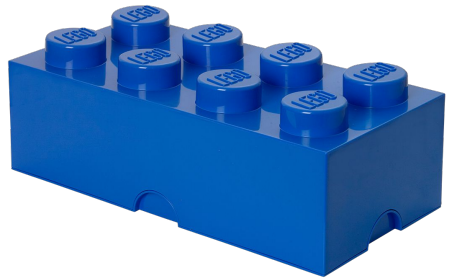
Desde Librerías

Conjuntos de funciones y módulos preescritos que facilitan tareas comunes. Permiten ahorrar tiempo al reutilizar código probado, en lugar de escribirlo todo desde cero.

PyTorch o TensorFlow

Desde Librerías

```python
def initialize_parameters():
    """Initialize network parameters (weights and biases)."""
new *
def forward_propagation():
    """Perform forward propagation to compute activations."""
new *
def compute_cost():
    """Compute the cost (loss) after forward propagation."""
new *
def backward_propagation():
    """Compute gradients for backpropagation."""
new *
def update_parameters():
    """Update network parameters using computed gradients."""
```



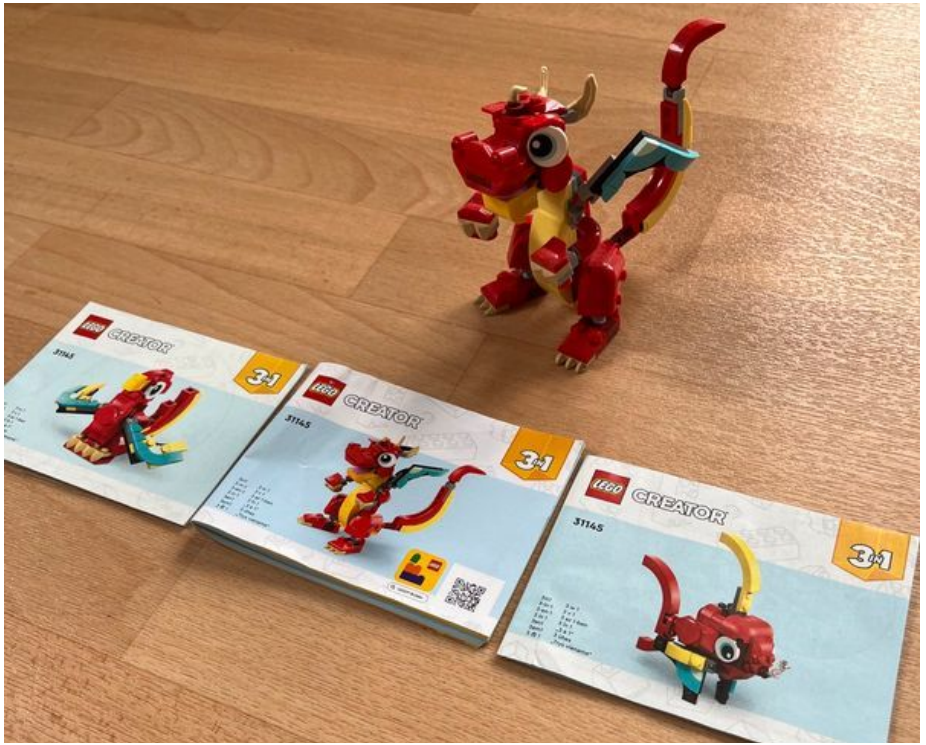
```python
new *
def activation_function():
    """Implement the activation function (e.g., ReLU, sigmoid, tanh)."""
new *
def activation_derivative():
    """Compute the derivative of the activation function for backpropagation."""
new *
def initialize_velocity():
    """Initialize velocity vectors for optimization with momentum."""
new *
def initialize_adam():
    """Initialize parameters for the Adam optimization algorithm."""
```

```python
new *
def optimize():
    """Execute the optimization process (e.g., SGD, momentum, Adam)."""
new *
def plot_cost():
    """Plot the cost function over time to visualize training progress."""
```

EDteam Curso EDteam desde cero

Desde Librerías

Con los mismos bloques se pueden crear diferentes arquitecturas



Pros

Acelerar el desarrollo. Se usan soluciones listas para usar.

Código **probado** y **optimizado** por la comunidad.

Simplifican el código final.

Contras

EDteam

Limitan la **personalización** de algunos detalles

Se **depende de terceros** para actualizaciones y correcciones

Desde Frameworks de alto nivel

Diseño como capas enteras, no tanto como bloques

Keras o fast.ai

Pros

La solución está **prácticamente hecha**

Probado y **optimizado** por la comunidad.

Contras

Limitan casi toda la personalización

Se **depende totalmente de terceros** para actualizaciones y correcciones

Se requiere un **alto conocimiento** en las Librerías asociadas al framework

EDteam

Desde Modelos

Arquitecturas de redes neuronales comunes ya hechas

SKLearn

Pros

Soluciones **hechas**, pocas líneas de código

Probado y **optimizado** por la comunidad.

Contras

EDteam

Limitan toda la personalización de las arquitecturas

Se **depende totalmente de terceros** para actualizaciones y correcciones

Se requiere un **alto conocimiento** en Librerías, frameworks y Teoría de las redes

En resumen...

Se pueden crear arquitecturas de redes neuronales desde 0, con Librerías o con Frameworks

Por regla general, a mayor Flexibilidad, menor Simplicidad en la arquitectura y las capacidades de la Red