Cheat Sheet

PyTorch

Deep learning with PyTorch

Kostya Numan
AI & AGI Researcher

PyTorch is an open-sou-rce machine learning library for deep learning and tensor computation with GPU acceleration.

Kostya Numan

AI & AGI Researcher

Tensors

Create a tensor from a list.

```
import torch
tensor = torch.tensor([1, 2, 3])
```

Create a random tensor.

```
torch.rand(3, 3)
```

Perform element-wise addition.

```
tensor + torch.ones(3)
```

Kostya Numan

AI & AGI Researcher

Automatic Differentiation

Enable gradients for a tensor.

```
x = torch.tensor(2.0, requires_grad=True)
```

Compute gradients.

```
y = x**2
y.backward()
```

Access gradient.

```
x.grad
```

Kostya Numan

AI & AGI Researcher

Neural Networks

Define a simple model.

```
import torch.nn as nn
model = nn.Linear(2, 1)
```

Apply model to input.

```
output = model(torch.tensor([1.0, 2.0]))
```

Access model weights.

```
model.weight.data
```

Kostya Numan

AI & AGI Researcher

Optimization

Define optimizer.

```
optimizer = torch.optim.SGD(
  model.parameters(), lr=0.01
)
```

Perform optimization step.

```
optimizer.step()
```

Zero gradients.

```
optimizer.zero_grad()
```

Kostya Numan

AI & AGI Researcher

Data Handling

Create a DataLoader.

```
from torch.utils.data import DataLoader
loader = DataLoader(
  dataset, batch_size=32
)
```

Iterate over DataLoader.

```
for batch in loader:
print(batch)
```

Kostya Numan

AI & AGI Researcher

Model Training

Train a model for one epoch.

```
for inputs, labels in loader:
  optimizer.zero_grad()
  outputs = model(inputs)
  loss = loss_fn(outputs, labels)
  loss.backward()
  optimizer.step()
```

Define loss function.

```
loss_fn = nn.MSELoss()
```

Kostya Numan

AI & AGI Researcher

Saving and Loading Models

Save the model state.

```
torch.save(
  model.state_dict(),
  'model.pth'
)
```

Load the model state.

```
model.load_state_dict(
  torch.load('model.pth')
)
```

Kostya Numan

AI & AGI Researcher

CUDA (GPU Acceleration)

Check for GPU availability.

```
torch.cuda.is_available()
```

Move tensor to GPU.

```
tensor = tensor.to('cuda')
```

Move model to GPU.

```
model = model.to('cuda')
```

Kostya Numan

AI & AGI Researcher

Subscribe

to Numan Substack
for more insights

Kostya Numan
AI & AGI Researcher