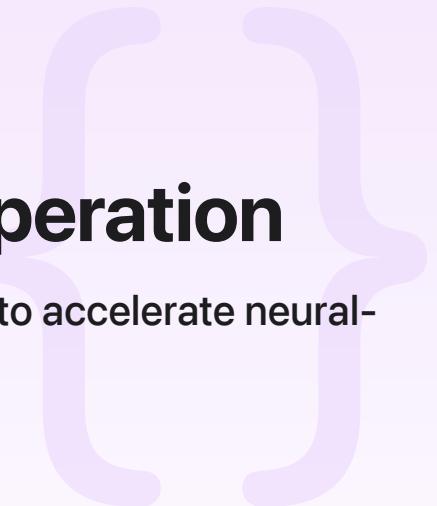


[Metal](#) / [Metal sample code library](#) / Customizing a TensorFlow operation

## Sample Code

# Customizing a TensorFlow operation

Implement a custom operation that uses Metal kernels to accelerate neural-network training performance.

[Download](#)

## Overview

### Note

This sample code project is associated with WWDC22 session [10063: Accelerate machine learning with Metal](#).

## Configure the sample code

1. Follow the instructions in [Getting started with tensorflow-metal](#).
2. Install ffmpeg using brew.

```
brew install ffmpeg
```

3. Install the required Python packages.

```
pip install -r requirements.txt
```

4. Use make to build the custom operation with Xcode.

```
cd hash_encoder  
make  
cd ..
```

## 5. Run the sample.

```
python tiny_nerf_hash.py
```

## 6. View the results in the result\_nerf\_hash folder.

- To compare the performance benefits provided by this sample, you can run the original NeRF sample code included with the project. View the results in the result\_nerf\_mlp folder.

```
python tiny_nerf_mlp.py
```

### Note

The sample uses low-resolution (100x100) images by default. You can alternatively use a high-resolution version of the data to produce a clearer rendering.

## See Also

### Compute workflows

- { } Performing calculations on a GPU  
Use Metal to find GPUs and perform calculations on them.
- { } Selecting device objects for compute processing  
Switch dynamically between multiple GPUs to efficiently execute a compute-intensive simulation.
- { } Customizing a PyTorch operation  
Implement a custom operation in PyTorch that uses Metal kernels to improve performance.