Daniel Wilczak, Jack Nguyen, Liam Kehoe, Nathan Foster

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

EasyNN Five Commandments.

1. Our Trained models will be EASY for users.
2. Making/Sharing your model should be EASY.
3. Image processing and accessibility - EASY.
4. Utilities to make life EASY.
5. Why we're better! (Or at least EASIER)

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

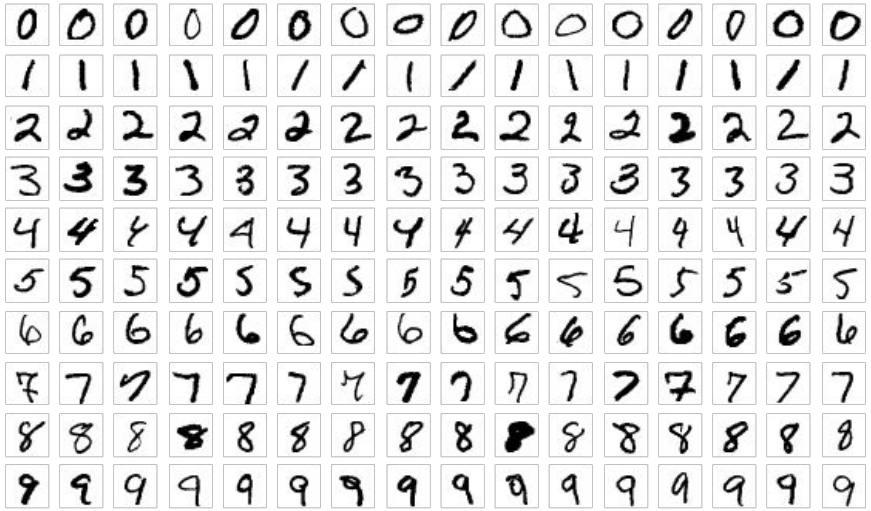
1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

Setup and classify in 30 seconds. (VIDEO)

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.



Dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

All you need - Easy.

```
from EasyNN.examples.mnist.number.trained import model

# Classify the an image in the dataset
print(model.classify(image))
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

Need a dataset example - Easy.

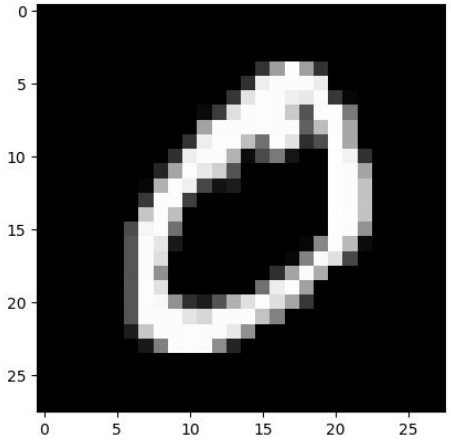
```
from EasyNN.examples.mnist.number.trained import model
from EasyNN.examples.mnist.number.data import dataset

images, labels = dataset

# Classify what the second image is in the dataset.
print(model.classify(images[1]))

# Show the image
model.show(images[1])
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

Dataset example output - Easy.




```
Downloading - number_parameters.npz:
[################################] 1769/1769 - 00:00:00
Downloading - number_structure.pkl:
[################################] 10700/10700 - 00:00:00
Downloading - number_dataset.npz:
[################################] 11221/11221 - 00:00:00
0
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

Download, Preprocess and Classify image from the internet - Easy.

```
from EasyNN.examples.mnist.number.trained import model
from EasyNN.utilities import Preprocess, download

# Download an example image.
download("three.jpg","https://bit.ly/3db01eV")

format_options = dict(
    grayscale=True,
    invert=True,
    process=True,
    contrast=30,
    resize=(28, 28),
    rotate=3,
)

# Converting your image into the correct format for the mnist number dataset.
image = Preprocess("three.jpg").format(**format_options)

# Classify what the image is using the pretrained model.
print(model.classify(image))

# Show the image after it has been processed.
model.show(image)
```

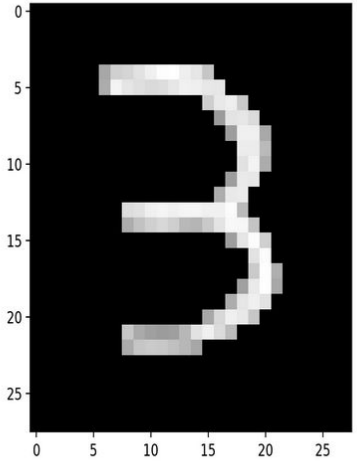
TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

Before:

After:



Result:

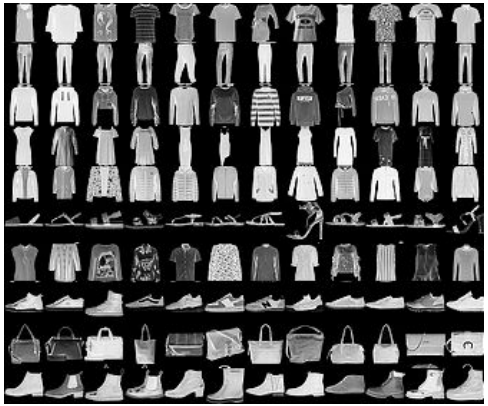
```
Downloading - four.jpg:
[################################] 1371/1371 - 00:00:00
3
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

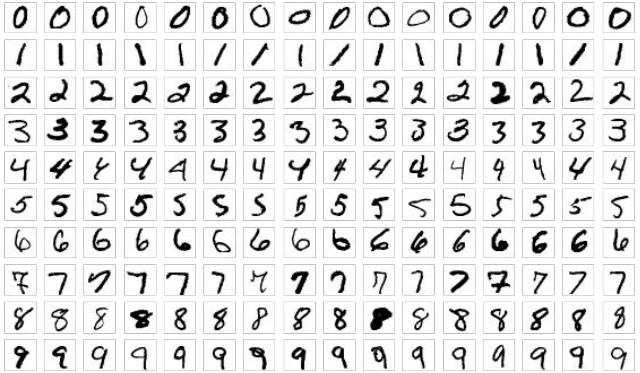
1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

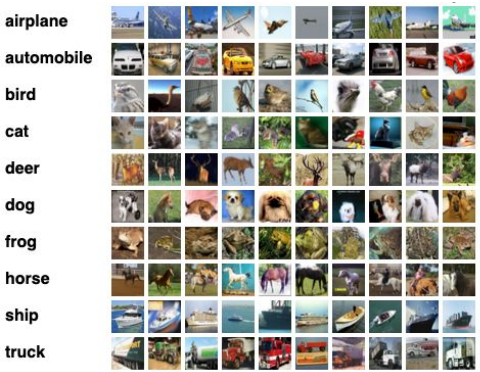
MORE TRAINED MODELS!



Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples.



Dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.



The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

1. Our Trained models will be EASY for users.

EASYNN
NEURAL NETWORKS MADE EASY

All our models use the same code - Easy

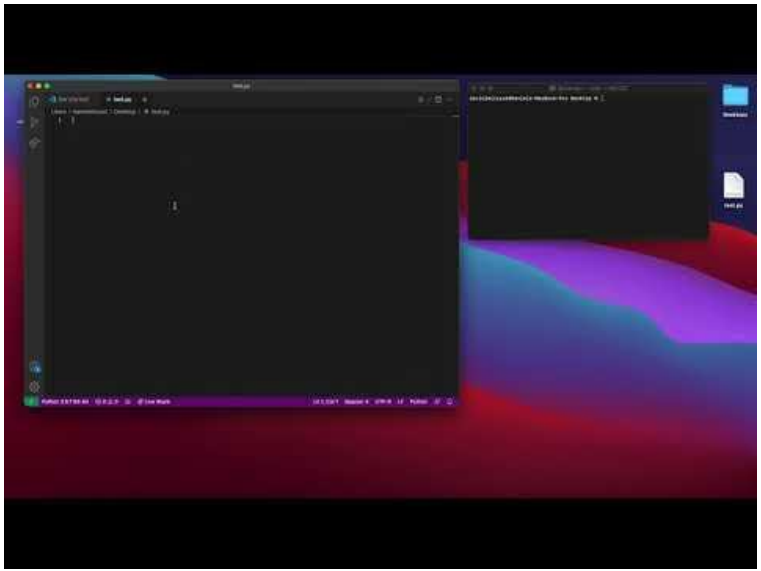
```
from EasyNN.examples.mnist.number.trained import model
from EasyNN.examples.mnist.fashion.trained import model
from EasyNN.examples.cifar10.trained import model
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Design, train, save and use in 30 seconds - (VIDEO)



TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Model Structure - Easy

```
# Create the mnist model.
model = Network(
    Normalize(1e-6),
    Randomize(0.3), 1024, ReLU,
    Randomize(0.2), 256, ReLU,
    Randomize(0.1), 256, ReLU,
    Randomize(0.03), 10,  LogSoftMax
)
```

```
# Create the Neural Network Model.
model = Network(
    16, ReLU,
    2, LogSoftMax,
)
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Setting Data and Labels - Easy

```
# Set the models labels
model.labels = {
    0: 0,
    1: 1
}
```

```
# Set your models data
model.training.data = (data, labels)
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Printing Training Progress - Easy

```
# Print the accuracy and iteration count every 10 iterations.
model.print.on_validation_start(iteration=True,accuracy=True)
model.print.on_training_start(iteration=True, frequency=10)
```

```
Iteration: 1780
Iteration: 1790
Iteration: 1800
Iteration: 1810
Iteration: 1820
Iteration: 1823, Validation Accuracy: 0.9130859375
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Stop training - Easy

```
# Always at the end of your setup
model.train()
```

Start Training - Easy

```
# Set when to terminate point. Training will end once your
# validation accuracy hits above 90% two times.
model.callback(
    cb.ReachValidationAccuracy(limit=0.90, patience=2),
)
```

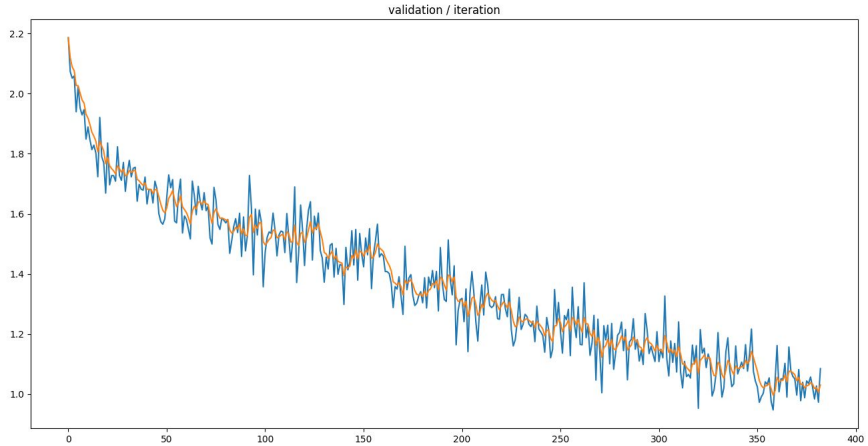
TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Plotting your progress - Easy

```
# Plot the progress.
model.callback(
    cb.PlotTrainingAccuracy(),
    cb.PlotValidationLoss(),
    cb.PlotTrainingAccuracy(),
    )
```



TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

2. Making/Sharing your model should be EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Saving Model - Easy

```
model.save("name")
```

Resulting Files:

```
name_parameters.npz
name_structure.pkl
```

Loading Model/Using - Easy

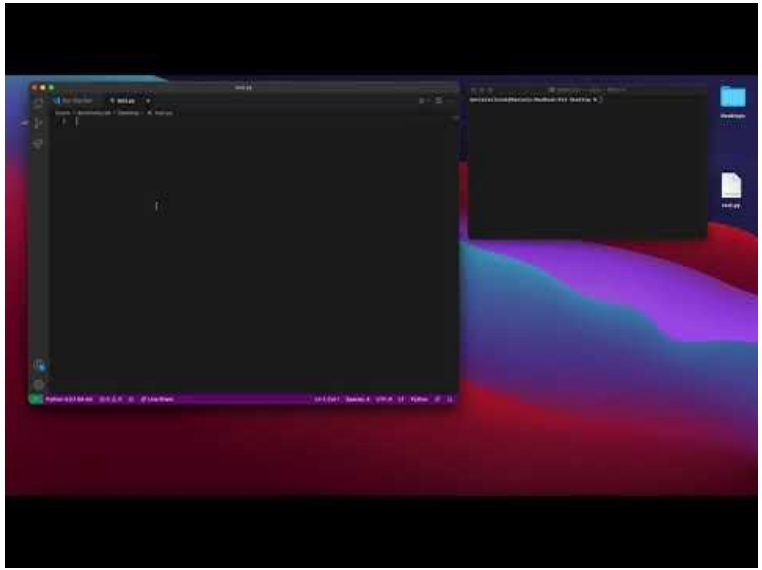
```
from EasyNN.model import Model

model = Model.load("name")
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

3. Image Processing and Accessibility - EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Image processing in from the internet in 30 seconds - (VIDEO)



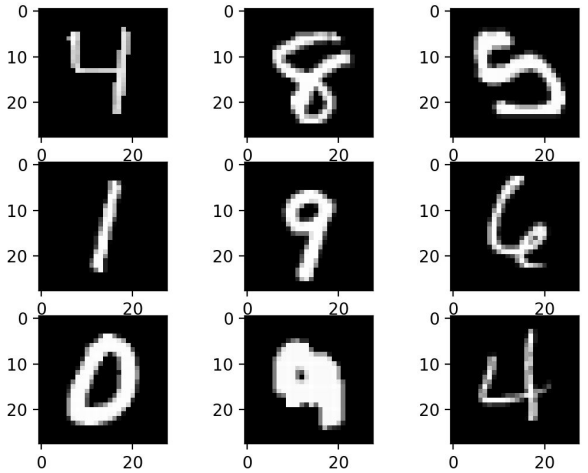
TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

3. Image Processing and Accessibility - EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Comparing your image and dataset - Easy

```
# Compare our image to random dataset images.
compare(image,dataset=dataset)
```



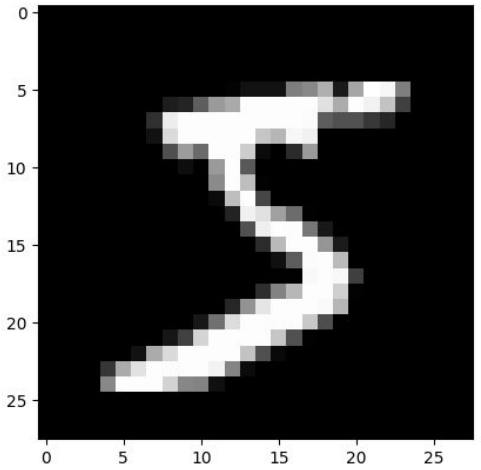
TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

3. Image Processing and Accessibility - EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Showing an image with dataset settings - Easy

```
from EasyNN.examples.mnist.number.trained import model
from EasyNN.examples.mnist.number.data import dataset

images, labels = dataset

model.show(images[0])
```



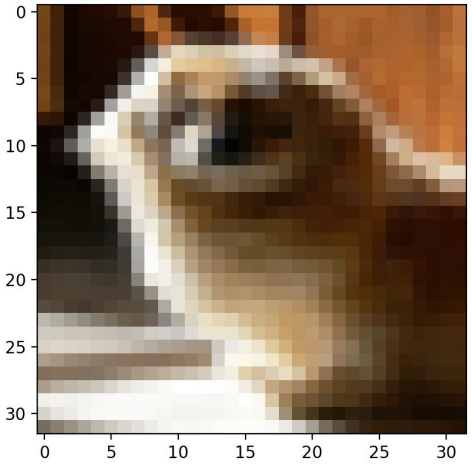
TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

3. Image Processing and Accessibility - EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Showing an image with dataset settings - Easy

```
from EasyNN.examples.cifar10.trained import model
from EasyNN.examples.cifar10.data import dataset

images, labels = dataset

# Show the image
model.show(images[2])
```

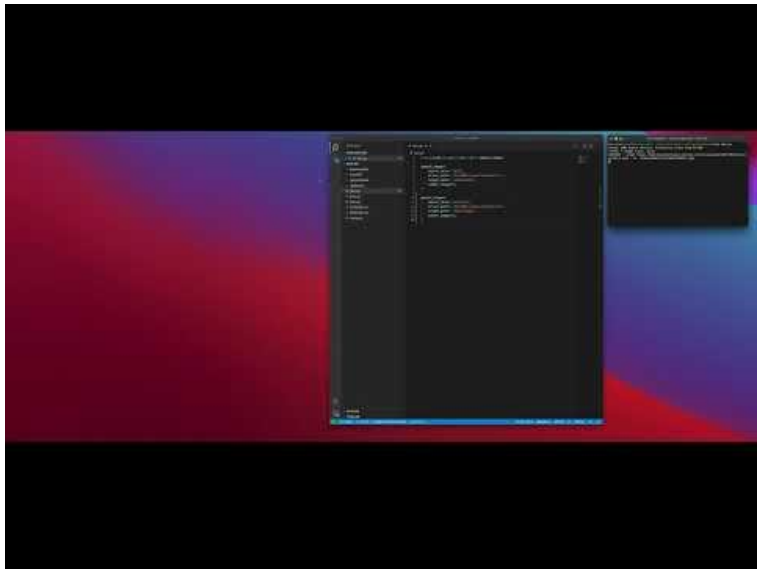


TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

4. Utilities to make life EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Web Scraping images from google - (VIDEO)



TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

4. Utilities to make life EASY.

EASYNN
NEURAL NETWORKS MADE EASY

Download your content from the internet - Easy

```
from EasyNN.utilities import download

# Save as file name / Url
download("dress.jpg","https://bit.ly/3b7rsXF")
```

```
Downloading - dress.jpg:
[################################] 25/25 - 00:00:00
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

5. Why we're better! (Or at least EASIER)

EASYNN
NEURAL NETWORKS MADE EASY

What makes us different? - The Competition

1. Code - Comparison with TensorFlow
2. Dependency and Drivers - Jax
3. GPU dependent - Pytorch
4. EasyNN Datasets in Numpy Format

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

5. Why we're better! (Or at least EASIER)

EASYNN
NEURAL NETWORKS MADE EASY

1. Code - Comparison with TensorFlow

EasyNN

```
from EasyNN.examples.mnist.number.trained import model
from EasyNN.examples.mnist.number.data import dataset

images, labels = dataset

# Classify what the second image is in the dataset.
print(model.classify(images[0]))

# Show the image.
model.show(images[0])
```

TensorFlow

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

5. Why we're better! (Or at least EASIER)

EASYNN
NEURAL NETWORKS MADE EASY

2. Dependency and Drivers - Jax

EasyNN

Nothing but Python and pip.

JAX

- Doesn't require gpu or hardware permissions.
- People with windows struggles with JAX because of permissions and drivers.
- Maybe some people don't have gpu

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

5. Why we're better! (Or at least EASIER)

EASYNN
NEURAL NETWORKS MADE EASY

3. GPU dependent - Pytorch

EasyNN	PyTourch
CPU BASED	CPU and GPU required.

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

5. Why we're better! (Or at least EASIER)

EASYNN
NEURAL NETWORKS MADE EASY

4. EasyNN Datasets in Numpy Format

```
from EasyNN.examples.mnist.number.data import dataset

# Dataset given in easy to use numpy arrays.
images, labels = dataset

print(images[0])
```

```
[##############################] 11221/11221 - 00:00:01
[  0   0   0   0   0   ... 253 253 253 244 133  ...   0   0]
```

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN

Questions?

TRY IT! **Github:** https://bit.ly/31qmy72 **Python:** Pip3 install EasyNN