RESNEXT WSL

By Facebook AI

ResNext models trained with billion scale weakly-supervised data.

View on Github Open on Google Colab

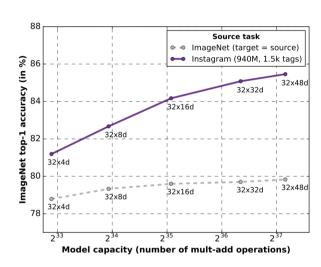


Fig. 5: Classification accuracy on val-IN-1k using ResNeXt-101 32×{4, 8 16, 32, 48}d with and without pretraining on the IG-940M-1.5k dataset.

```
import torch
model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x8d_wsl')
# or
# model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x16d_wsl')
# or
# model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x32d_wsl')
# or
#model = torch.hub.load('facebookresearch/WSL-Images', 'resnext101_32x48d_wsl')
model.eval()
```

All pre-trained models expect input images normalized in the same way, i.e. mini-batches of 3-channel RGB images of shape $(3 \times H \times W)$, where H and W are expected to be at least 224. The images have to be loaded in to a range of [0, 1] and then normalized using mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225].

Here's a sample execution.

```
# Download an example image from the pytorch website
import urllib
url, filename = ("https://github.com/pytorch/hub/raw/master/dog.jpg", "dog.jpg")
try: urllib.URLopener().retrieve(url, filename)
except: urllib.request.urlretrieve(url, filename)
```

```
# sample execution (requires torchvision)
from PIL import Image
from torchvision import transforms
input_image = Image.open(filename)
preprocess = transforms.Compose([
   transforms.Resize(256),
   transforms.CenterCrop(224),
   transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
input_tensor = preprocess(input_image)
input_batch = input_tensor.unsqueeze(0) # create a mini-batch as expected by the model
# move the input and model to GPU for speed if available
if torch.cuda.is_available():
    input_batch = input_batch.to('cuda')
    model.to('cuda')
with torch.no_grad():
    output = model(input_batch)
# Tensor of shape 1000, with confidence scores over Imagenet's 1000 classes
print(output[0])
# The output has unnormalized scores. To get probabilities, you can run a softmax on it.
print(torch.nn.functional.softmax(output[0], dim=0))
```

Model Description

The provided ResNeXt models are pre-trained in weakly-supervised fashion on **940 million** public images with 1.5K hashtags matching with 1000 ImageNet1K synsets, followed by fine-tuning on ImageNet1K dataset. Please refer to

To analyze traffic and optimize your experience, we serve cookies on this site. By clicking or navigating, you agree to allow our usage of cookies. As the current maintainers of this site, Facebook's Cookies Policy applies. Learn more, including about available controls: Cookies Policy.

ResNeXt-101 32x16d	193M	36B	84.2	97.2
ResNeXt-101 32x32d	466M	87B	85.1	97.5
ResNeXt-101 32x48d	829M	153B	85.4	97.6

Our models significantly improve the training accuracy on ImageNet compared to training from scratch. We achieve state-of-the-art accuracy of 85.4% on ImageNet with our ResNext-101 32x48d model.

References

• Exploring the Limits of Weakly Supervised Pretraining

Docs

Access comprehensive developer documentation for PyTorch

View Docs

Tutorials

Get in-depth tutorials for beginners and advanced developers

View Tutorials

Resources

Find development resources and get your questions answered

View Resources

PyTorch Resources

Get Started Tutorials

Features Docs
Ecosystem Discuss

Blog GitHub Issues

Slack

Contributing

Stay Connected

Email Address