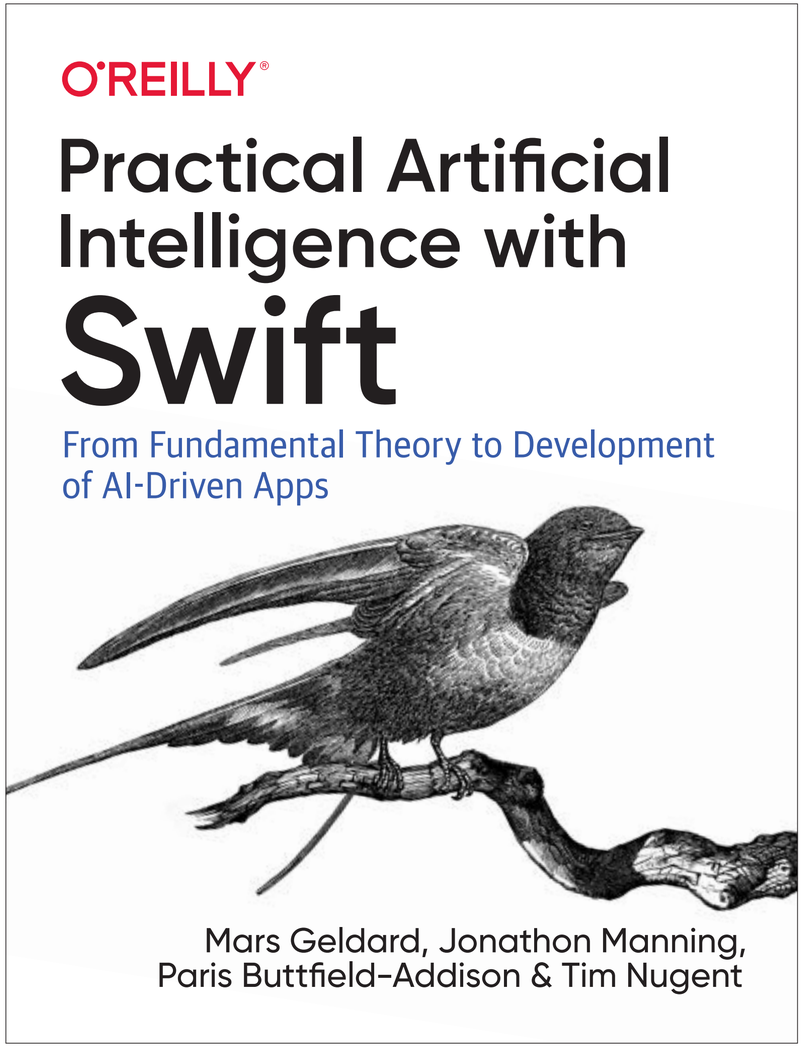
A quick look at

On-device Artificial Intelligence

Neural Style Transfer

with Mars Geldard &
Dr Paris Buttfield-Addison


O'REILLY®
Practical Artificial Intelligence with
Swift
From Fundamental Theory to Development of AI-Driven Apps
Mars Geldard, Jonathon Manning,
Paris Buttfield-Addison & Tim Nugent

@parisba

@themartianlife

Neural Style Transfer

Neural Style Transfer

(Image A Content + Image B Style = Image C)

This isn’t an Apple talk

We just know a lot about their technology stack.

Task-based AI/ML


turi
create intelligence™

https://github.com/apple/turicreate


ML

I want to...	Machine Learning Task
Label Images	Image Classification
Recognize objects within images	Object Detection
Find similar images	Image Similarity
Create stylized avatars / profile images	Style Transfer
Personalize choices for users	Recommender
Detect an activity using sensors	Activity Classification
Analyze sentiment of messages	Text Classifier
Predict a label	Classifiers
Predict numeric values	Regression
Group similar datapoints together	Clustering

Part 1

Neural Style Transfer

Part 2

Demonstration

Part 1

Neural Style Transfer

“Style Transfer is a task wherein the stylistic elements of a style image are imitated onto a new image while preserving the content of the new image.”

–TuriCreate documentation

What is NST?

Technically, what’s going on?

Neural Network refresher

Using a model to generate?


KELETI PALY

Solved problem?

Why is this hard on-device?

Quick and efficient

We need it to the be fast.

The Old Way

Using a neural network… but not really training.

An Old Way

- Trained classifier as a feature detector: texture synthesis
  - e.g. using a network to extract features from a texture image.
- Checking how close the synthesised image is to a content image with respect to the features extracted by the classifier
- Repeat.
- Feedforward network, going from content to new styled image in one pass, but only works on a single style.

It’s always the same problem…

As it turns out!

A Neural Algorithm of Artistic Style

Leon A. Gatys,[1,2,3*] Alexander S. Ecker,[1,2,4,5] Matthias Bethge[1,2,

[1]Werner Reichardt Centre for Integrative Neuroscience
and Institute of Theoretical Physics, University of Tübingen, Germany
[2]Bernstein Center for Computational Neuroscience, Tübingen, Germany
[3]Graduate School for Neural Information Processing, Tübingen, German
[4]Max Planck Institute for Biological Cybernetics, Tübingen, Germany
[5]Department of Neuroscience, Baylor College of Medicine, Houston, TX,
*To whom correspondence should be addressed; E-mail: leon.gatys@bethge

In fine art, especially painting, humans have mastered the skill to create
visual experiences through composing a complex interplay between
tent and style of an image. Thus far the algorithmic basis of this p
unknown and there exists no artificial system with similar capabiliti
ever, in other key areas of visual perception such as object and face re
near-human performance was recently demonstrated by a class of b
inspired vision models called Deep Neural Networks.[1,2] Here we in
artificial system based on a Deep Neural Network that creates art
of high perceptual quality. The system uses neural representati
rate and recombine content and style of arbitrary images, provid
algorithm for the creation of artistic images. Moreover, in ligh
ing similarities between performance-optimised artificial neural
biological vision,[3–7] our work offers a path forward to an algo
standing of how humans create and perceive artistic imagery.

Published as a conference paper at ICLR 2017

A Learned Representation for Artistic Style

Vincent Dumoulin & Jonathon Shlens & Manjunath Kudlur
Google Brain, Mountain View, CA
vi.dumoulin@gmail.com, shlens@google.com, keveman@google.com

Abstract

The diversity of painting styles represents a rich visual vocabulary for the construction of an image. The degree to which one may learn and parsimoniously capture this visual vocabulary measures our understanding of the higher level features of paintings, if not images in general. In this work we investigate the construction of a single, scalable deep network that can parsimoniously capture the artistic style of a diversity of paintings. We demonstrate that such a network generalizes across a diversity of artistic styles by reducing a painting to a point in an embedding space. Importantly, this model permits a user to explore new painting styles by arbitrarily combining the styles learned from individual paintings. We hope that this work provides a useful step towards building rich models of paintings and offers a window on to the structure of the learned representation of artistic style.

1 Introduction

A pastiche is an artistic work that imitates the style of another one. Computer vision and more recently machine learning have a history of trying to automate pastiche, that is, render an image in the style of another one. This task is called *style transfer*, and is closely related to the texture synthesis task. While the latter tries to capture the statistical relationship between the pixels of a source image which is assumed to have a stationary distribution at some scale, the former does so while also attempting to preserve some notion of content.

On the computer vision side, Efros & Leung (1999) and Wei & Levoy (2000) attempt to "grow" textures one pixel at a time using non-parametric sampling of pixels in an examplar image. Efros & Freeman (2001) and Liang et al. (2001) extend this idea to "growing" textures one patch at a time, and Efros & Freeman (2001) uses the approach to implement "texture transfer", i.e. transfering the texture of an object onto another one. Kwatra et al. (2005) approaches the texture synthesis problem from an energy minimization perspective, progressively refining the texture using an EM-like algorithm. Hertzmann et al. (2001) introduces the concept of "image analogies": given a pair of "unfiltered" and "filtered" versions of an examplar image, a target image is processed to create an analogous "filtered" result. More recently, Frigo et al. (2016) treats style transfer as a local texture transfer (using an adaptive patch partition) followed by a global color transfer, and Elad & Milanfar (2016) extends Kwatra's energy-based method into a style transfer algorithm by taking content similarity into account.

On the machine learning side, it has been shown that a trained classifier can be used as a feature extractor to drive texture synthesis and style transfer. Gatys et al. (2015a) uses the VGG-19 network (Simonyan & Zisserman, 2014) to extract features from a texture image and a synthesized texture. The two sets of features are compared and the synthesized texture is modified by gradient descent so that the two sets of features are as close as possible. Gatys et al. (2015b) extends this idea to style transfer by adding the constraint that the synthesized image also be close to a content image with respect to another set of features extracted by the trained VGG-19 classifier.

While very flexible, this algorithm is expensive to run due to the optimization loop being carried. Ulyanov et al. (2016a), Li & Wand (2016) and Johnson et al. (2016) tackle this problem by introducing a *feedforward style transfer network*, which is trained to go from content to pastiche image in one pass. However, in doing so some of the flexibility of the original algorithm is lost: the style transfer network is tied to a single style, which means that separate networks have to be trained

The New Way

The New Way

“Hey computer, learn the style of this picture, and later I’ll give you some other pictures to make look like this. Thanks!”

The New Image

Activations in the neural network that are similar
to both the style and to the content activations.

A million approaches

- Turi Create + Core ML + Swift iOS app
- Keras + Core ML Tools + Core ML + Swift iOS app
- PyTorch + Core ML Tools + Core ML + Swift iOS app
- TensorFlow + Core ML Tools + Core ML + Swift iOS app
- …. and many others!


turi
create intelligence™

Part 1

Neural Style Transfer

Part 2

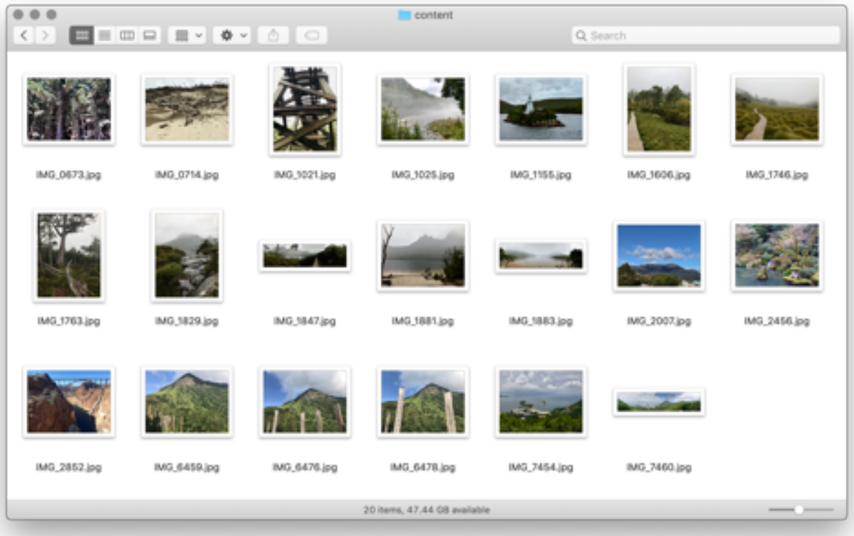
Demonstration

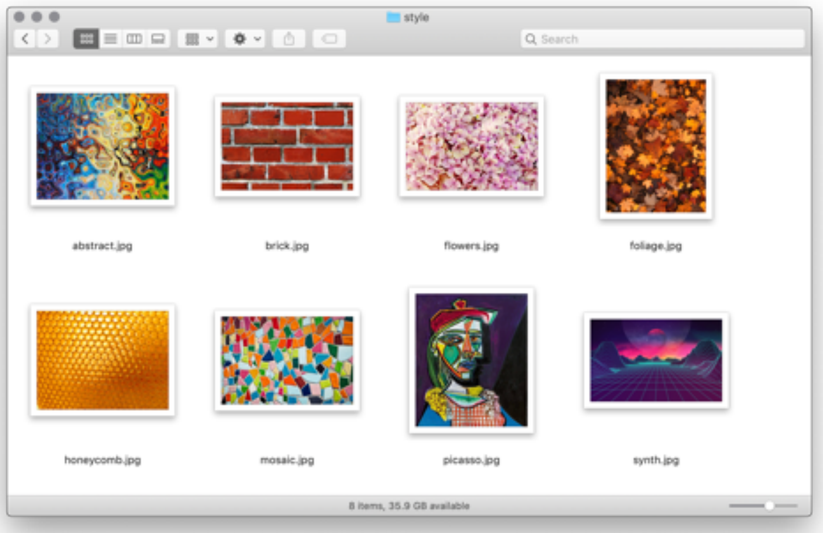
Part 2

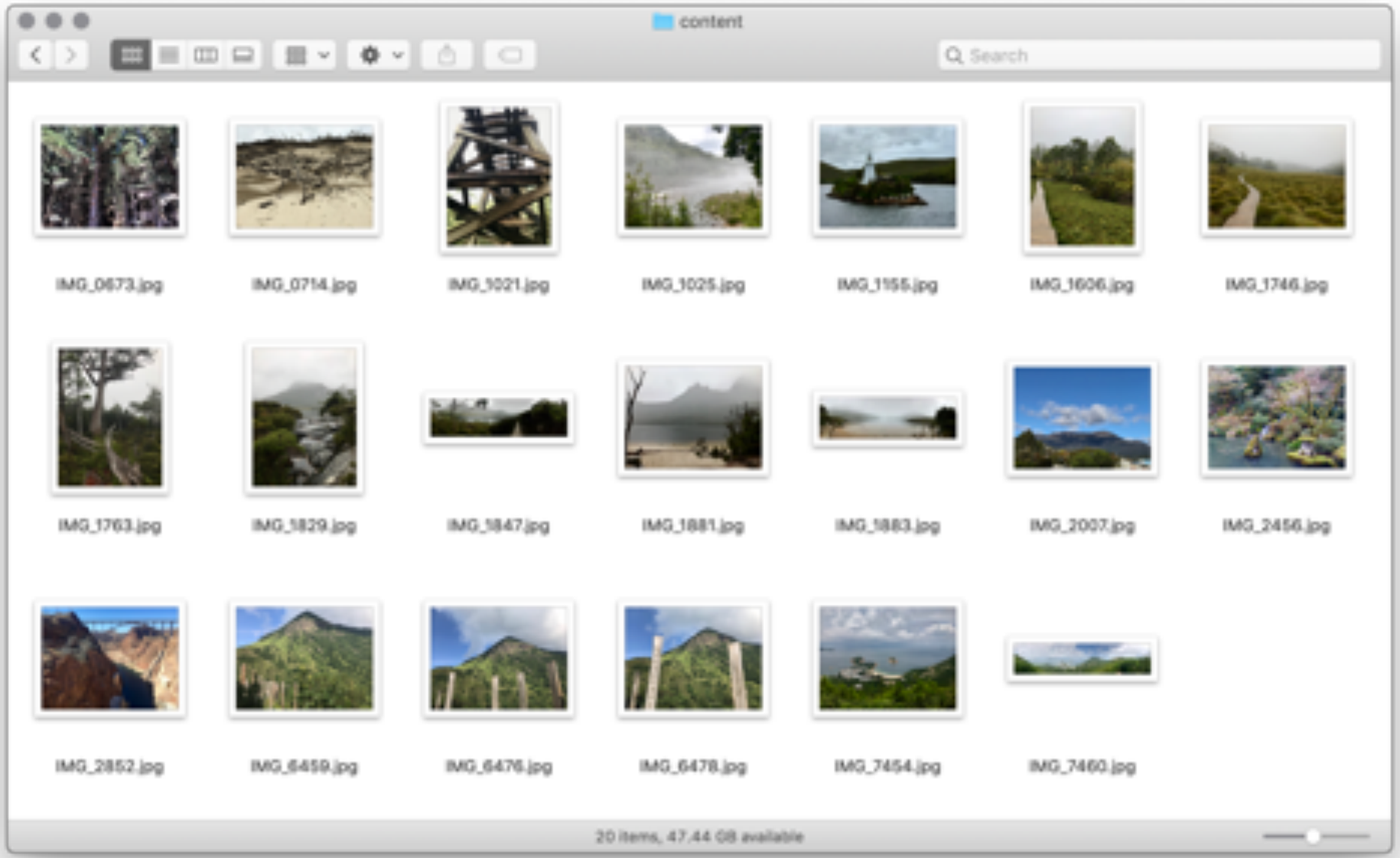
Demonstration

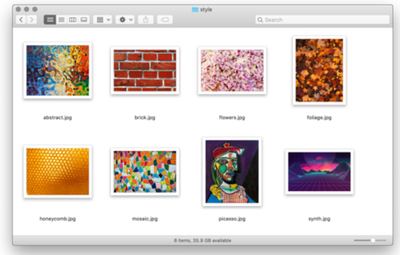
Data Preparation

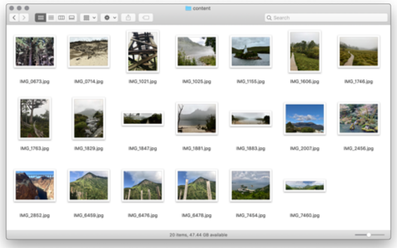
Step 1

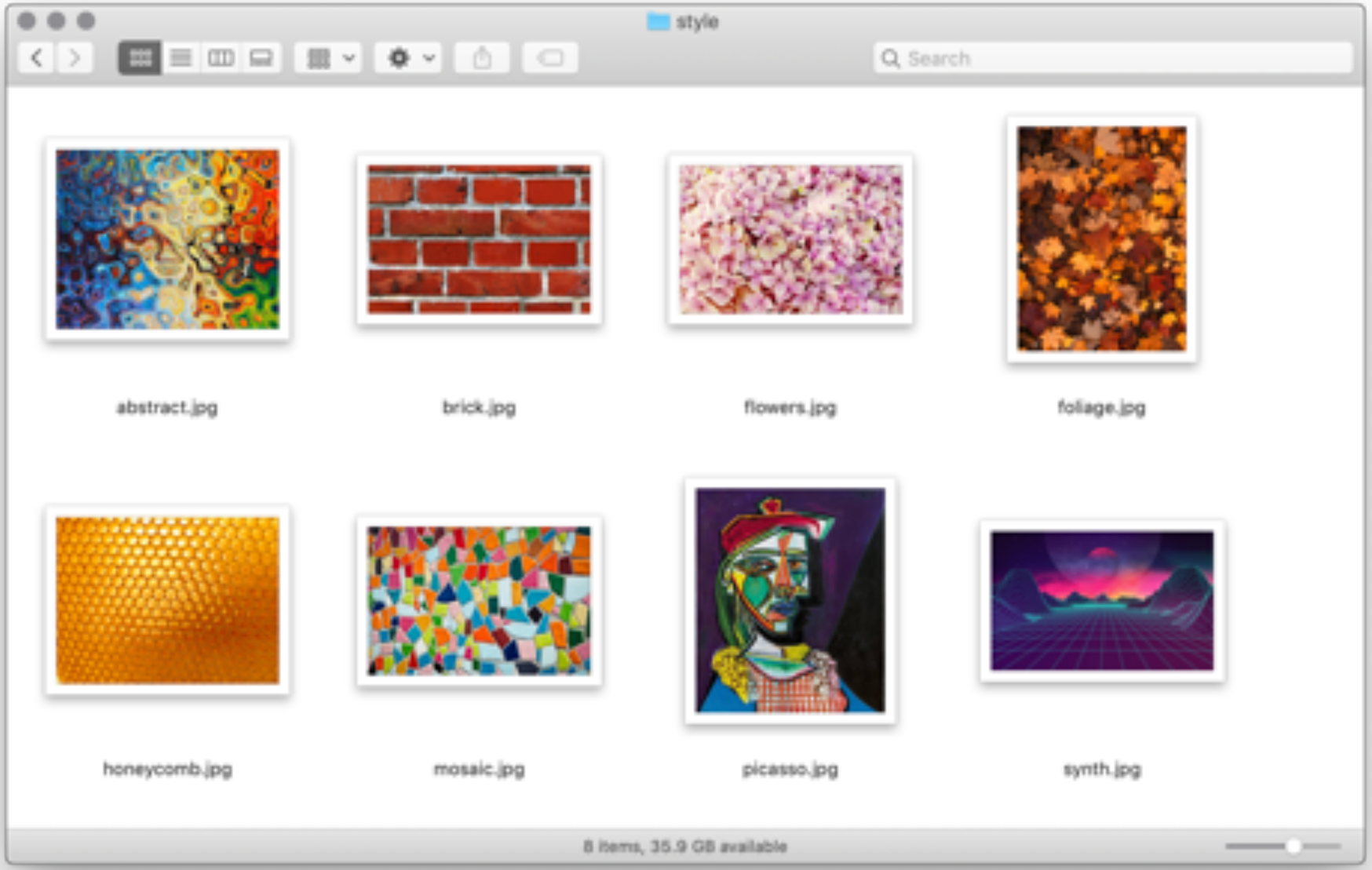

content
Search
IMG_0673.jpg
IMG_0714.jpg
IMG_1021.jpg
IMG_1025.jpg
IMG_1155.jpg
IMG_1606.jpg
IMG_1746.jpg
IMG_1763.jpg
IMG_1829.jpg
IMG_1847.jpg
IMG_1881.jpg
IMG_1883.jpg
IMG_2007.jpg
IMG_2456.jpg
IMG_2852.jpg
IMG_6459.jpg
IMG_6476.jpg
IMG_6478.jpg
IMG_7454.jpg
IMG_7460.jpg
20 items, 47.44 GB available

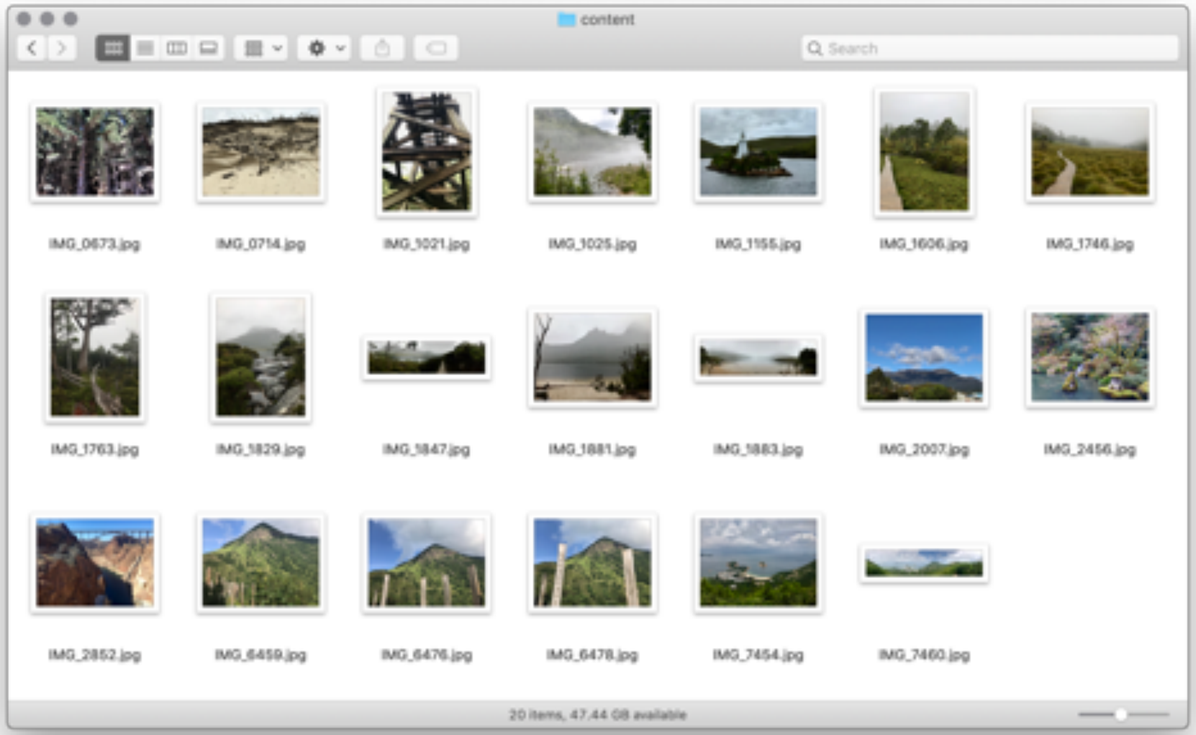

style
Search
abstract.jpg
brick.jpg
flowers.jpg
foliage.jpg
honeycomb.jpg
mosaic.jpg
picasso.jpg
synth.jpg
8 items, 35.9 GB available

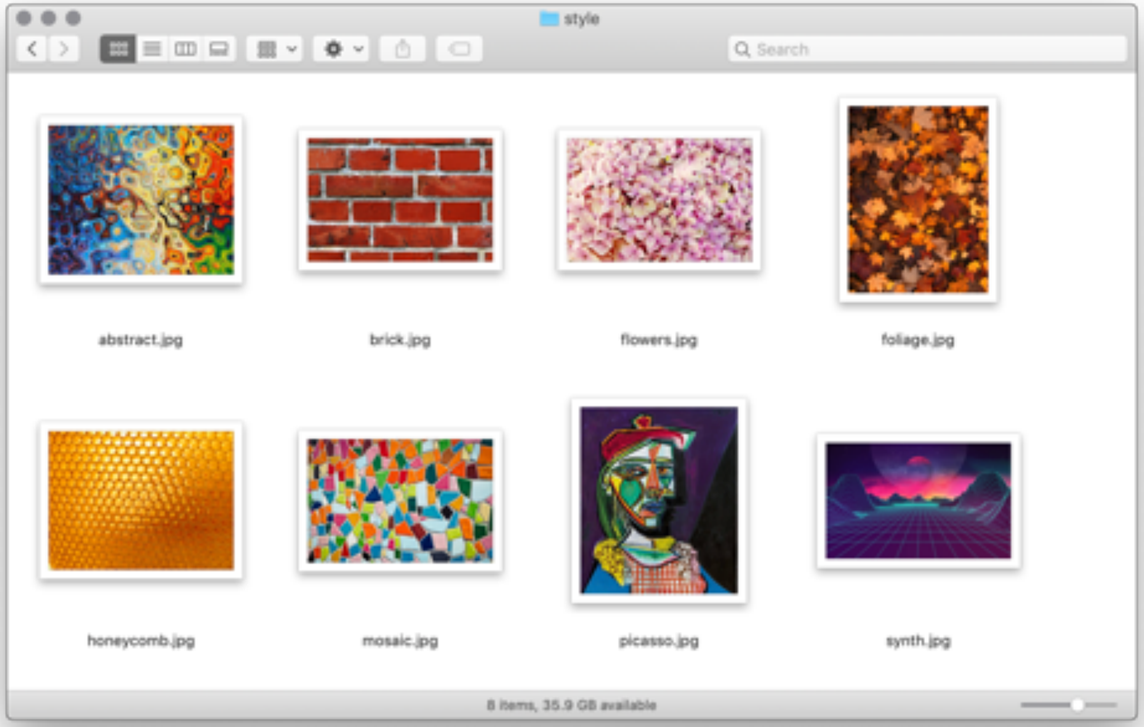

content
Search
IMG_0673.jpg
IMG_0714.jpg
IMG_1021.jpg
IMG_1025.jpg
IMG_1155.jpg
IMG_1606.jpg
IMG_1746.jpg
IMG_1763.jpg
IMG_1829.jpg
IMG_1847.jpg
IMG_1881.jpg
IMG_1883.jpg
IMG_2007.jpg
IMG_2456.jpg
IMG_2852.jpg
IMG_6459.jpg
IMG_6476.jpg
IMG_6478.jpg
IMG_7454.jpg
IMG_7460.jpg
20 items, 47.44 GB available





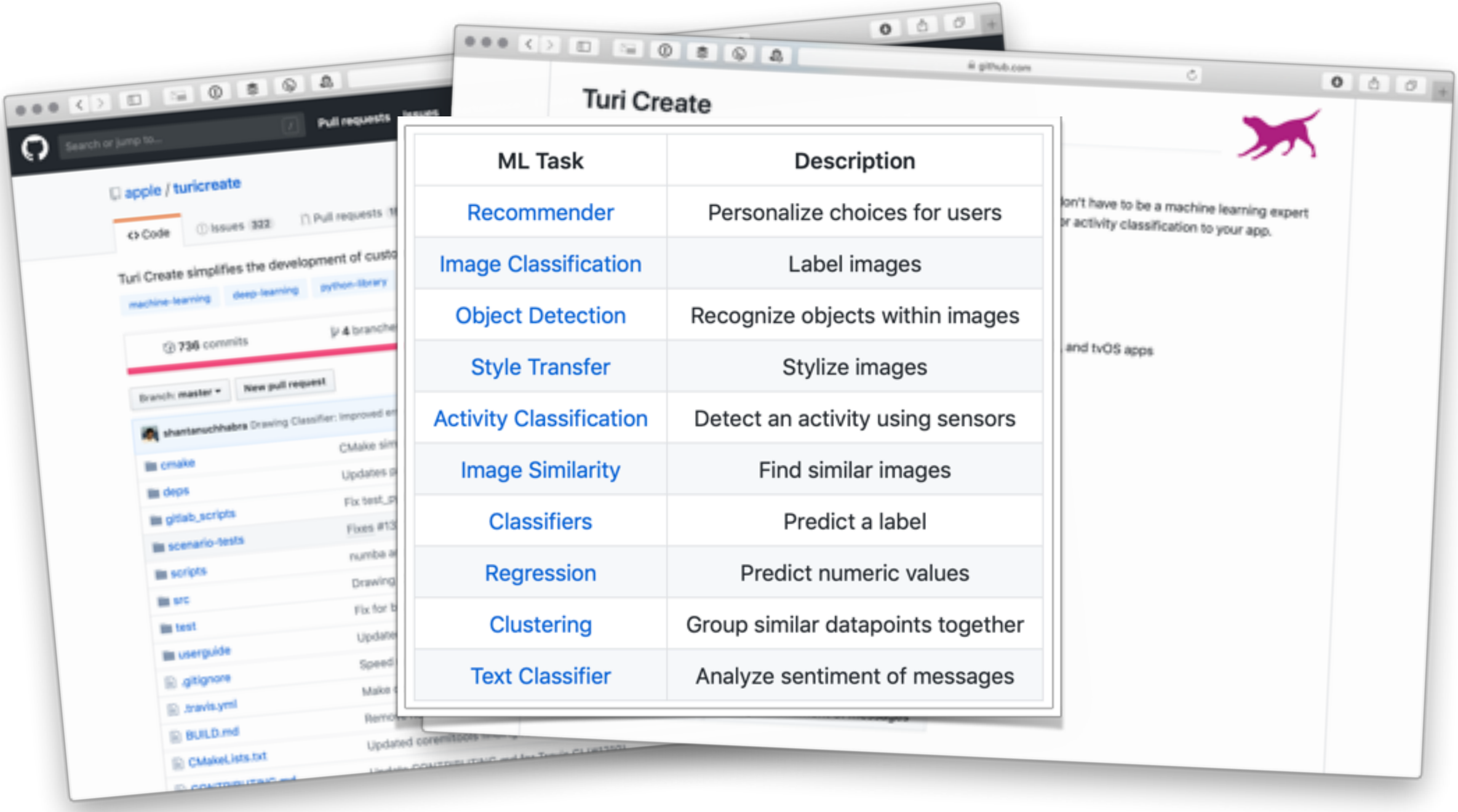

style
Search
abstract.jpg
brick.jpg
flowers.jpg
foliage.jpg
honeycomb.jpg
mosaic.jpg
picasso.jpg
synth.jpg
8 items, 35.9 GB available


content
Search
IMG_0673.jpg
IMG_0714.jpg
IMG_1021.jpg
IMG_1025.jpg
IMG_1155.jpg
IMG_1606.jpg
IMG_1746.jpg
IMG_1763.jpg
IMG_1829.jpg
IMG_1847.jpg
IMG_1881.jpg
IMG_1883.jpg
IMG_2007.jpg
IMG_2456.jpg
IMG_2852.jpg
IMG_6459.jpg
IMG_6476.jpg
IMG_6478.jpg
IMG_7454.jpg
IMG_7460.jpg
20 items, 47.44 GB available


style
Search
abstract.jpg
brick.jpg
flowers.jpg
foliage.jpg
honeycomb.jpg
mosaic.jpg
picasso.jpg
synth.jpg
8 items, 35.9 GB available

Training

Step 2


Turi Create
apple / turicreate
Turi Create simplifies the development of custo
machine-learning
deep-learning
python-library
Code
Issues 332
Pull requests
736 commits
4 branches
Branch: master
New pull request
cmake
deps
gitlab_scripts
scenario-tests
scripts
src
test
userguide
.gitignore
.travis.yml
BUILD.md
CMakeLists.txt
ML Task
Description
Recommender
Personalize choices for users
Image Classification
Label images
Object Detection
Recognize objects within images
Style Transfer
Stylize images
Activity Classification
Detect an activity using sensors
Image Similarity
Find similar images
Classifiers
Predict a label
Regression
Predict numeric values
Clustering
Group similar datapoints together
Text Classifier
Analyze sentiment of messages
and tvOS apps

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```

```
import turicreate as tc

# Load the style and content images
styles = tc.load_images('style/')
content = tc.load_images('content/')

# Create a StyleTransfer model
model = tc.style_transfer.create(styles, content, max_iterations=2000)

# Load some test images
test_images = tc.load_images('test/')

# Stylize the test images
stylized_images = model.stylize(test_images)

# Save the model for later use in Turi Create
model.save('DemoNST.model')

# Export for use in Core ML
model.export_coreml('DemoNST.mlmodel')
```


turicreate-training — train.py — 80×24
(venv) Hyperion:turicreate-training desplesda$ ./train.py


NST — python train_nst.py — python — python • python train_nst.py — 119×30
python train_nst.py
Using 'image' in style_dataset as feature column and using 'image' in content_dataset as feature column
Downloading https://docs-assets.developer.apple.com/turicreate/models/resnet-16.params
Download completed: /var/folders/sz/9p_l7vsn4zn0f4051by7yh9m0000gn/T/model_cache/resnet-16.params
Downloading https://docs-assets.developer.apple.com/turicreate/models/vgg16-conv1_1-4_3.params
Download completed: /var/folders/sz/9p_l7vsn4zn0f4051by7yh9m0000gn/T/model_cache/vgg16-conv1_1-4_3.params
1	20.058	7.5
2	19.391	29.7
3	22.495	52.2
4	25.156	74.5
5	24.308	97.4
6	23.609	126.4
7	23.020	153.1
8	25.688	179.0
9	24.263	203.8
10	22.451	230.1
11	21.359	260.5
12	20.486	292.7
13	19.633	319.2
14	18.834	342.8
15	21.639	367.2
16	20.758	392.9
17	23.566	418.9
18	22.874	444.5

Training

Training

Super fast

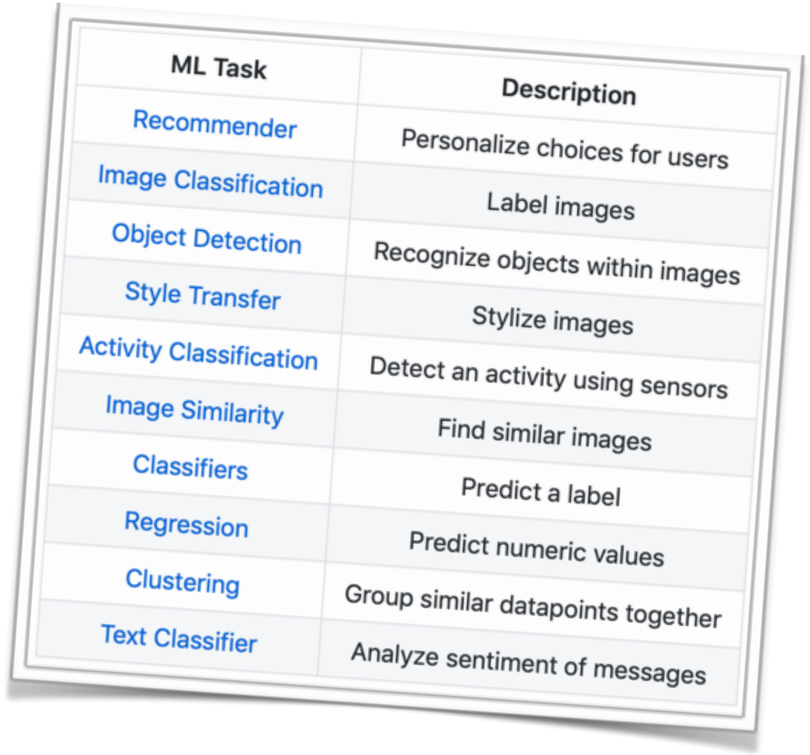
Quick & Easy to Make & Deploy

Maybe not so quick to run…

Demo

Thank You!

Task-based AI/ML



ML Task	Description
Recommender	Personalize choices for users
Image Classification	Label images
Object Detection	Recognize objects within images
Style Transfer	Stylize images
Activity Classification	Detect an activity using sensors
Image Similarity	Find similar images
Classifiers	Predict a label
Regression	Predict numeric values
Clustering	Group similar datapoints together
Text Classifier	Analyze sentiment of messages

Task-based AI/ML

Great resources

(You should check them out)

- https://developer.apple.com/videos/play/wwdc2018/712/
- https://github.com/apple/turicreate/
- https://github.com/alexsosn/iOS_ML
- https://www.manning.com/books/grokking-deep-learning
- https://medium.com/artists-and-machine-intelligence/neural-artistic-style-transfer-a-comprehensive-look-f54d8649c199

Stay in touch

- Tweet at us 🐣
- Stalk us on the internet:
  - www.paris.id.au
  - www.themartianlife.com
- Say g'day if you see us around this week 👋
- Slides + code 👉 github.com/thesecretlab/ReinforceConf2019