

From Data to Web Application: Anime Character Image Recognition with Transfer Learning

PyCon JP 2018 #pyconjp #pyconjp_1



Iskandar Setiadi Software Engineer at HDE, Inc.



Github

https://github.com/freedomofkeima

Blog https://freedomofkeima.com/blog/

Speaker in:

- <u>PyCon Japan 2017</u> (last year!)
- PyCon Indonesia 2017
- PyCon Malaysia 2018



Iskandar Setiadi

From Jakarta, Indonesia Software Engineer at Japan

HDE, Inc. (https://hde.co.jp/en/)

Image Recognition in Real Life



Self-driving car

Smart home

Image Recognition in Daily Life





Smart workplace

Face ID

ILSVRC

Largest Computer Vision Competition

Starting from 2015, deep learning has better top-5 error score compared to human (1000 categories)!

Case #2: ILSVRC 2010-2015





Int J Comput Vis (2015) 115:211-252



Image Recognition Challenges

Background







Problem

- Cropped images
- Edited images
- Unindexed images

	- 1 - 1	
→ G Secure https://saucenao.com/sea	ircn.pnp	
56% funded	SauceNA	O Needs Your Support, Upgrade or Donate Today!
Sufficiency	Illumination Pixiv ID: 61127204 @ Member: tucana @	
P Patreon re need your help! Donate	Creator: tucana Material: love live! school idol project love live! sunshine!! love live! (series)	29.23% Characters: sakurauchi riko nishikino maki
Low similarity results h	ave been hidden. Click here to display them	
\leftarrow \rightarrow \mathcal{C} \triangleq Secure https://saucenao.com/		
56% funded	SauceN	IAO Needs Your Support, Upgrade or Donate Today
Sauce NAD	Shanimuni Go Shanimuni Go - v16 c91 [batoto] - (Manga)	42.10
● Patreon We need your help! Donate	雪燐オンリー兄さんといっしよ【新刊】 Pixiv ID: 23599975 @ Member: 碧乃魅沙@雪燐オンリーE12	· 雪燐合同誌:週末監禁 41.9** 変

What is Machine Learning?





Deep Learning

Traditional ML:

Increasing number of training iterations will eventually get stagnated at certain point.

Alternative proposal:

More layers! But it is slow :(



Deep Learning: Convolution



Image



Convolved Feature

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0



Visualization of a curve detector filter

Deep Learning



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Step 1. Preparation

TensorFlow Installation

\$ pip3 install --upgrade tensorflow
or
\$ pip3 install --upgrade tensorflow-gpu

URL: <u>https://www.tensorflow.org/install/</u>

Why should we use Transfer Learning?

From certain Top-5 characters indexing website:

- 35000 registered characters
- Top 1000 characters: 70+ images
- Top 2000 characters: 40+ images

Dataset size is small! Google Inception-v3 uses > 1000 images per category.

With transfer learning, we don't need to retrain low-level features extraction model.

Result: **100 categories, ~ 60 images per category**.

URL: <u>https://www.tensorflow.org/tutorials/image_retraining</u>

Scrap & clean up the data



Note: Please contact me if you want to know the details

Google Dataset Search (Beta)

Secure https://toolbox.google.com/datasetsearch



About

Google Dataset Search Beta

Search for Datasets

Q

Try boston education data or weather site:noaa.gov

Google Dataset Search

Q anime chara

\times

About

....

Feec

kaggle Tagged Anime Illustrations www.kaggle.com

Updated Jul 30, 2018

kaggle Anime Recommendations Database www.kaggle.com

Updated Dec 21, 2016

kaggle The Simpsons Characters Data www.kaggle.com

Updated Apr 13, 2018

ArtiosCAD Workspace File vistablog.com.ua

Danbooru2017: A large-scale crowdsourced and tagged anime illustration dataset

The first set of data comes from the imageboard Danbooru. The entire corpus of Danbooru images was scraped from the site with permission and was collected into a dataset. The zip files included here have the full metadata for these images as well as a subset of 300,000 of the images in normalized 512px x 512px form. Full information about this dataset is available here:

https://www.gwern.net/Danbooru2017

From the article:

Deep learning for computer revision relies on large annotated datasets. Classification/categorization has benefited from the creation of ImageNet, which classifies 1m photos into 1000 categories. But classification/categorization is a coarse description of an image which limits application of classifiers, and there is no comparably large dataset of images with many tags or labels which would allow learning and detecting much richer information about images. Such a dataset would ideally be >1m images with at least 10 descriptive tags each which can be publicly distributed to all interested researchers, hobbyists, and organizations. There are currently no such public datasets, as ImageNet, Birds, Flowers, and MS COCO fall short either on image or tag count or restricted distribution. I suggest that the image - boorus be used. The image boorus are longstanding web databases which host large numbers of images which can be tagged or labeled with an arbitrary number of textual descriptions; they were developed for and are most popular among fans of anime, who provide detailed annotations.

The best known booru, with a focus on quality, is Danbooru. We create & provide a torrent which contains ~1.9tb of 2.94m images with 77.5m tag instances (of 333k defined tags, ~26.3/image) covering Danbooru from 24 May 2005 through 31 December 2017 (final ID: #2,973,532), providing the image files & a JSON export of the metadata. We also provide a smaller torrent of SFW images downscaled to 512x512px JPG (241GB; 2,232,462 images) for convenience.

Our hope is that a Danbooru2017 dataset can be used for rich large-scale classification/tagging & learned embeddings, test out the transferability of existing computer vision techniques (primarily developed using photographs) to illustration/anime-style images, provide an archival backup for the Danbooru community, feed back metadata improvements & corrections, and serve as a testbed for advanced techniques such as conditional image generation or style transfer.

anime character

Step 2. Analyze the Problem

Face Detection: Introduction



Face Detection (Human Face)

Adapted from https://github.com/shantnu/FaceDetect:

import cv2

faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

```
image = cv2.imread(imagePath)
gray = cv2.cvtColor(image, cv2.COLOR BGR2GRAY)
```

Face Detection (Human Face)



Face Detection (Same Model - Anime Face)



2D is **Better** not equal to 3D face!

Facial features are different!

e.g.: 2D has no nose

		* 19	20	21			* 2	*24	*25 *	26	
	*18			* 24	2			- -		*27	
		*3	* 38 * 7* 42*	39 41 ^{**} 40		*28	16	* 44 * 43 * 48	* 45 * 47 [*] 46		
* 1						*29					*17
						* 30					* 10
*2						*31					* 10
*3				9	* 32* 3	3* 34*	35 36	3			* 15
*4				* 50	* 51 * 62	* 52 * 63	* 53 * 64	* 54			*14
				* 49* 61	* 68	*67	* 66	* 56			* 13
	*5				* 59	* 58	*5/				
		*6								* 12	
			*7						*11		
				*8		¥ 9		*10			

Face Detection: Train New Model!

Adapted from https://github.com/nagadomi/lbpcascade animeface:

import cv2

```
cascade = cv2.CascadeClassifier('lbpcascade animeface.xml")
```

```
image = cv2.imread(imagePath)
gray = cv2.cvtColor(image, cv2.COLOR BGR2GRAY)
```

Face Detection (Anime Face)



Face Detection (Same Model - Human Face)



Face Recognition

Face Detection \rightarrow "Accomplished"

Full-layered Deep Learning \rightarrow Requires a huge dataset, weeks to train

Google Inception-v3: 1.2 million training data, 1000 classes, 1 week to train

Step 3. Train & Validate the Model

What does the layers learn?





Transfer Learning for Anime Characters

Transfer Learning: Retrained Layers

Dropout: Dropping out units to prevent overfitting

Fully Connected: Extracting global features, every node in the layer is connected to the preceding layer

Softmax: Squashing final layer to make a prediction, which sums up to 1. For example, if we have 2 classes and class A has the value of 0.95, then class B will have the value of 0.05.

Transfer Learning: Retrain Final Layer

Build the retrainer:

\$ bazel build tensorflow/examples/image_retraining:retrain

Execute the retrainer:

\$ bazel-bin/tensorflow/examples/image_retraining/retrain --image_dir ~/images

Hyperparameters: learning rate, number of iterations, distortions factor, ...

Types of headache



Original ImageNet: Accuracy vs Dataset size

From 10 to 100 training images per class, you can improve your Top-1 accuracy by 5%.

From 100 to 1000 training images per class, you can only improve it by less than 2%.

Low number of images \rightarrow bad accuracy

Huge number of images \rightarrow stagnated accuracy



URL: https://annals-csis.org/Volume_12/drp/pdf/526.pdf

Step 4. Serve as a Web App

MoeFlow: Specification

 \rightarrow Build with Sanic (Flask-like Python 3.5+ web server); **Caveat**: Good for prototyping, have some memory handling issues

 \rightarrow While training model requires huge GPU resources (g2.2xlarge), using retrained model can be hosted in server with small resources (t2.small)

What it does:

- Run face detection with OpenCV
- Resize image to a fixed proportion
- Run classification with TensorFlow

MoeFlow: Web App

```
@app.listener('before server start')
async def initialize(app, loop):
    moeflow_path = os.environ.get('MOEFLOW_MODEL_PATH')
    label path = os.path.join(os.sep, moeflow path, "output labels 2.txt")
    model path = os.path.join(os.sep, moeflow path, "output graph 2.pb")
    app.label lines = [
        line.strip() for line in tf.gfile.GFile(label_path)
    graph = tf.Graph()
    graph def = tf.GraphDef()
    with tf.gfile.FastGFile(model_path, 'rb') as f:
        graph def.ParseFromString(f.read())
    with graph.as default():
        tf.import_graph_def(graph_def, name='')
    app.graph = graph
    logging.info("MoeFlow model is now initialized!")
```

URL: https://github.com/freedomofkeima/MoeFlow

```
def classify_resized_face(file_name, label_lines, graph):
    results = []
    logging.info('Processing classification')
    with tf.Session(graph=graph) as sess:
        # Feed the image data as input to the graph and get first prediction
        softmax_tensor = sess.graph.get_tensor_by_name('final_result:0')
        input_operation = sess.graph.get_operation_by_name("Mul")
        t = read tensor from image file(file name)
        predictions = sess.run(
            softmax_tensor,
            {input operation.outputs[0]: t}
        )
        # Sort to show labels of first prediction in order of confidence
        top_k = predictions[0].argsort()[-3:][::-1]
        for node id in top k:
            human string = label lines[node id]
            score = predictions[0][node_id]
            results.append((human_string, score))
```

```
return results
```

2017-12-06 01:51:53 - (network)[INF0][127.0.0.1:49154]: GET http://127.0.0.1:888 8/static/images/83a97094d7634333b5c9d7e8e0901da4.jpg 200 119582 INF0:network: 2017-12-06 01:51:53 - (network)[INF0][127.0.0.1:49156]: GET http://127.0.0.1:888 8/static/images/006410fcf9114fbb89a4f32ede402056 ing 200 5002
INF0:network:
2017-12-06 05:48:33 - (network)[INF0][127.0.0.1:59920]: GET http://127.0.0.1:888 8/ 200 1180
INF0:network:
INFO:root:Height: 1416, Width: 1280
INF0:root:Input file is created at /tmp/tmpgm8odxy2.jpg
INFO:root:Processing classification
INF0:root:[('tedeza rize', 0.8941825), ('yui', 0.015284972), ('takanashi rikka', 0.014930859)]
2017-12-06 05:49:38 - (network)[INF0][127.0.0.1:59970]: POST http://127.0.0.1:88
88/ 200 1565
INF0:network:
2017-12-06 05:49:39 - (network)[INF0][127.0.0.1:59972]: GET http://127.0.0.1:888
8/static/images/fda2e4333b194f30a845ed8d320bf449.jpg 200 294605
INF0:network:
2017-12-06 05:49:39 - (network)[INF0][127.0.0.1:59974]: GET http://127.0.0.1:888
8/static/images/d9a530d76c17404tbdbde7a2e5bf9c55.jpg 200 5807 INF0:network:

MoeFlow: Use Retrained Model



Input:



Output:

Character

Prediction (Top-3)



- kafuu chino
- miki sayaka
- kashima

Results / Demo

Input:



Output:

Character



Prediction (Top-3)

- jouga maya
- hyoudou michiru
- ujimatsu chiya
- kafuu chino
- izumi sagiri
- aqua
- <u>y</u>
- natsu megumi
- yoshino
- inokuma youko

Results / Demo

Input: rotation / axis problem

Output:

Character



- Prediction (Top-3)
 - suzukaze aoba
 - kafuu chino
 - yamada elf
 - yagami kou
 - ayase eli
 - tomoe mami

Results / Demo

Other Projects / Alternatives

rezoo/illustration2vec

Semantic feature vectors.

For example:

- Character

- General (hair color, eye color, etc)

A simple deep learning library for estimating a set of tags and extracting semantic feature vectors from given illustrations.

32 comm	nits 🖗 1	branch	🗞 1 release	🤽 1 con	tributor		ক MIT
Branch: master 🕶	New pull request			Create new file	Upload files	Find file	Clone or download *
🔞 rezoo Update RE	ADME.md				Lates	st commit 8	F4b11d on Aug 30, 2017
i2v	Merge brand	h 'master' of gith	nub.com:rezoo/illustration2vec				3 years ago
🖿 images	Add an exam	iple image					3 years ago
papers	Add papers						2 years ago
.gitignore	Add .gitigno	e					2 years ago
	add LICENSE						3 years ago
README.md	Update READ	OME.md					a year ago
get_models.sh	Update READ	OME.md					a year ago

Illustration2Vec

illustration2vec (i2v) is a simple library for estimating a set of tags and extracting semantic feature vectors from given illustrations. For details, please see our main paper.

Chainer + illustration2vec

\leftrightarrow \rightarrow C \blacksquare Secure https://chainer-colab-n	otebook.readt	hedocs.io/ja/latest/notebook/hands_on_ja/chainer/classify_anime_characters.html#	
🖨 Chainer Colab Notebook			
latest	In []:	%matplotlib inline	
		<pre>import matplotlib.pyplot as plt</pre>	
Search docs		from PIL import Image	
		from chainer import cuda	
show on		chainer.config.train = False	
		for _ in range(10):	
Colaboratory		<pre>x, t = valid[np.random.randint(len(valid))]</pre>	
		<pre>x = cuda.to_gpu(x)</pre>	
		<pre>y = F.softmax(model.predictor(x[None,]))</pre>	
学羽川 ― プを書いてっ トう		pred = os.path.basename(dnames[int(v.data.argmax())])	
子宮ルーノを言いしみよう		label = os.path.basename(dnames[t])	
Trainerを使ってみよう			
新しいネットワークを書いてみよう		print('pred:', pred, 'label:', label, pred == label)	
データセットクラスを書いてみよう		x = cuda.to_cpu(x)	
Chainerハンズオン: Pythonによろディ		x = x / 256	
ープラーニング入門		x = np.clip(x, 0, 1)	
		<pre>plt.imshow(x.transpose(1, 2, 0))</pre>	
Chainer Tutorial Bookへようこそ!		plt.show()	
Classify Anime Characters with Fine- tuning Model		pred: 168_asagiri_mai label: 168_asagiri_mai True	
ChainerRL で atari のゲームを DON で		0	
解かしてみる		20	
ChainerRL クイックスタートガイド		40	
Casting and Applicate with Descuring		60	
Neural Network		80	
Synthesize Human Speech with			
WaveNet			
Word2Vec: Obtain word embeddings		140	
1 Internation			

50 75 100 125 150

PaaS - Platform as a Service

- Clarifai \rightarrow since 2016

- Azure Custom Vision \rightarrow Beta release as 2018

- Google Cloud AutoML \rightarrow Beta release as 2018



aws Services v Resource Groups 🗸 \$ 🛆 🛛 Iskandar Setiadi 👻 N. Virginia 💌 Support * Amazon Rekognition Face comparison Metrics Compare faces to see how closely they match based on a similarity percentage. Demos InvalidParameterException (400) 0 Object and scene detection Request has Invalid Parameters (Image must contain detectable faces) Image moderation Facial analysis Reference face Comparison faces Done with the demo? Celebrity recognition Learn more Face comparison Results Text in image Request Video Demos Response Video analysis Additional Resources Getting started guide Download SDKs Developer resources Pricing FAQ Forum Choose a sample image Choose a sample image

Clarifai UI



PRECISION 0.75 0.92 1.00 0.86 0.67 1.00 0.90 1.00 1.00 0.80



Clarifai: Python API

Read the Docs

← → C 🔒 Secure https://clarifai	-python.readthedocs.io/en/latest/tutorial/#upload-images
🖨 clarifai latest	Tutorial
Search docs	Each of the examples below is a small independent code snippet within 10 lines that could work by copy and paste to a python source code file. By playing with them, you should be getting started with Clarifai API. For more information about the API, check the API Reference.
Installation Guide	Predict Tutorial
🗉 Tutorial	Predict with Public Models
Predict Tutorial	Feedback lutorial Concept model prediction
Feedback Tutorial	Detection model prediction
Upload Images	Face detection model prediction
Create a Model Train the Model	Upload Images
Predict with Model Instantiate an Image Bulk Import Images Search the Image	<pre>from clarifai.rest import ClarifaiApp app = ClarifaiApp() app = clarifaiApp() app.inputs.create_image_from_url(url='https://samples.clarifai.com/puppy.jpeg', concepts=['my pupp app.inputs.create_image_from_url(url='https://samples.clarifai.com/puppy.jpeg', not_concepts=['my baseline="concenter">concepts=['my pupp baseline="concenter">concepts=['my pupp baseline="concenter">concepts=['my pupp baseline="concenter") baseline="concenter">concepts=['my pupp baseline="concenter") baseline="concenter" baseline="concenter") baseline="concenter" baseline="concenter" baseline="concenter" baseline="concenter") baseline="concenter" baseline="concenter" baseline="concenter" baseline="concenter") baseline="concenter" baseline="concente</pre>
Basic Concepts	(
REFERENCES	
API Reference	Create a Model
	Note
	This assumes you follow through the tutorial and finished the "Upload Images" Otherwise you may not be able to create the model.
	<pre>1 model = app.models.create(model_id="puppy", concepts=["my puppy"])</pre>
	Train the Model

Google Cloud AutoML: Python API

	AutoML Vision BETA	MoeFlow	+ ADD IMAGES +	II, LABEL STATS	EXPORT DATA	iskandar-test-project 👻
=	predict.py					
õ	import sys					
	from google.cloud import from google.cloud.automl	automl_v1beta1 _v1beta1.proto :	import service_pb2			
	<pre>def get_prediction(conte prediction_client = au</pre>	nt, project_id, toml_v1beta1.Pre	model_id): edictionServiceClien	t()		
	<pre>name = 'projects/{}/lo payload = {'image': {'</pre>	cations/us-cents	rall/models/{}'.form	at(project_id, mod	el_id)	
	params = {}	Indge_bytes : Co	Succue II			
	request = prediction c	lient.predict(na	ame, payload, params)		
	return request # wait	s till request :	is returned			
	ifname == 'main	1.				
	file_path = sys.argv[1]				
	<pre>project_id = sys.argv[</pre>	2]				
	<pre>model_id = sys.argv[3]</pre>					
	with open(file_path, '	rb') as ff:				
	<pre>content = ff.read()</pre>					
	<pre>print get_prediction(c</pre>	ontent, project	_id, model_id)			

Execute the request

python predict.py YOUR LOCAL IMAGE FILE iskandar-test-project ICN5640832962537657205

Quick Comparison (as August 2018)

Top-1 Accuracy with pre-processed model (after face detection, same dataset)

Clarifai \rightarrow 95.2% (they have been in this business for years)

Google Cloud AutoML \rightarrow 93% (their model is based on improved version of Inception v3, so it fails on almost the same test data)

MoeFlow (my model) \rightarrow 88.6% (let be fair, I'm not a machine learning expert and my job is unrelated to machine learning \clubsuit , but at least it's free and I can put it together with my server!)

Azure Custom Vision → 77.4% (at least it's better than Microsoft, but they are still in Beta)

Clarifai

i_i



Details X ▼ TRANSFER-LEARNING-ANIME PREDICT C TRAIN 🗸 aragaki_ayase 0.59 Odekomori_sanae 0.00 🔿 akemi_homura 0.25 asuna 0.00 🔿 ayanami_rei 0.06 ayase_eli 0.00 asahina_mikuru 🔿 aqua 0.05 0.00 🔿 akiyama_mio 0.04 Oalice_cartelet 0.00 Create Concept ✓ GENERAL-V1.3 GENERAL-V1.3 illustration 0.97 face 0.90 0.96 man woman 0.90 fun 0.94 people 0.89 fashion 0.94 vector 0.89 a az music

0 00

...

vound

Google Cloud AutoML

	AutoML Vision BETA	MoeFlow	+ ADD IMAGES +	II, LABEL STATS	EXPORT DATA	iskandar-test-project 👻
≡ Q	Model MoeFlow_v20180803063827 Test your model on ne	∽ ew images				
	If your model will be used to make	e predictions on peop	ele, test your model on image	es that capture the diver	rsity of your userbase. Learn mo Predictions Only top 5 labels are aragaki_ayase akiyama_mio akemi_homura asuna asahina_mikuru	shown. 0.680 0.277 0.032 0.007 0.001

MoeFlow

Input:





Azure Custom Vision



My Tags aragaki_ayase Predictions Tag Probability aqua 11% 9.9% ayanami rei akemi_homura 8.8% akiyama_mio 6.3% aragaki_ayase 5.7% ayase_eli 2.1% 2% asuna asahina_mikuru 0.6%

When should we use PaaS for Image Recognition?

(+) Image recognition is not directly related to your main business and you don't want to invest too much time on it

(+) You don't need the capability of customizing training model

(+) You have money \$\$\$, PaaS tends to get more expensive in the long run

(+) You are satisfied with the result, of course, you should compare all available PaaS out there, they have their own strong points

What to compare:

- Single vs multi-objects, e.g.: Car vs [Car, Ball, ...]

- Single class vs multiple tags, e.g.: Car vs Toyota (Brand name), or [Vehicle: 0.99, Car: 0.97, ...]

- Top-1 vs Top-x accuracy

- Cost

Closing Remarks

"Never-ending" Development

As the number of category increases, Top-1 accuracy will also decrease since some characters are too similar (free-style art) and transfer learning effect is diminished.

- Image noise
- Rotation / axis
- Face expressions (closed eyes, etc)
- Characters with "multiple" forms
- Brightness & Contrast

Fooling Neural Network





Original image Output Label: Teapot





Generalizing Ideas

Solve Your Own Problems!



Github Projects

freedomofkeima/MoeFlow: Repository for anime characters recognition website (Alpha)

freedomofkeima/transfer-learning-anime: Transfer Learning for Anime Characters Recognition

freedomofkeima/opencv-playground: Compare 2D and 3D OpenCV Cascade Classifier

Presentation Slide https://freedomofkeima.com/pyconjp2018.pdf

Curated List https://github.com/kjw0612/awesome-deep-vision http://www.themtank.org/a-year-in-computer-vision

HDE, Inc. at Shibuya, Tokyo



Global Internship Program https://www.hde.co.jp/en/gip/

PyCon JP 2018 Sprint Sponsor



Full-timer benefits: Get your CfP accepted and the company will handle the rest!



Thank you!



