2D Anime Image Tag Detecting with Machine Learning

Ruozhou Zhang, X152437

timzhangrz98@gmail.com

Project Object

- In this project, I am focusing on giving a machine learning model for detecting 2D anime images.
- The expected outcome is that all the valid elements (contained in the element dictionary) will be recognized
- Develop Tools:

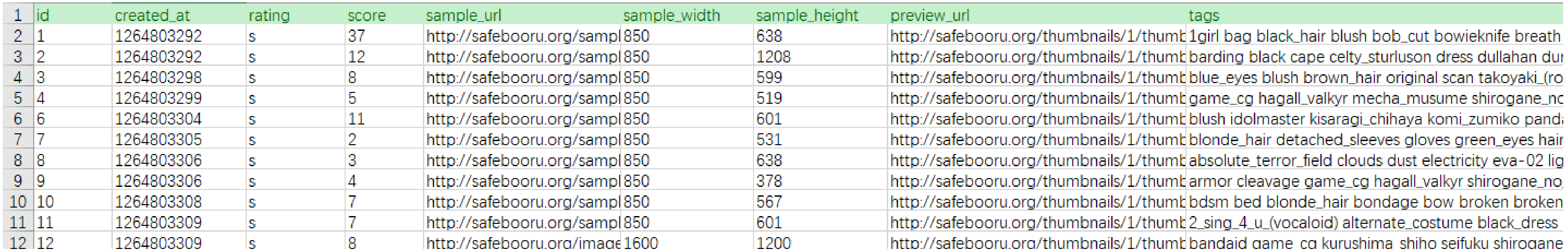
  Visual Studio Code

  TensorFlow 1.12.0

  Tensorboard 1.12.2

Raw Data

- Safebooru - Anime Image Metadata (Alexander Lamson, Kaggle) https://www.kaggle.com/alamson/safebooru/
- all_data.csv (more than 1,900,000 data)

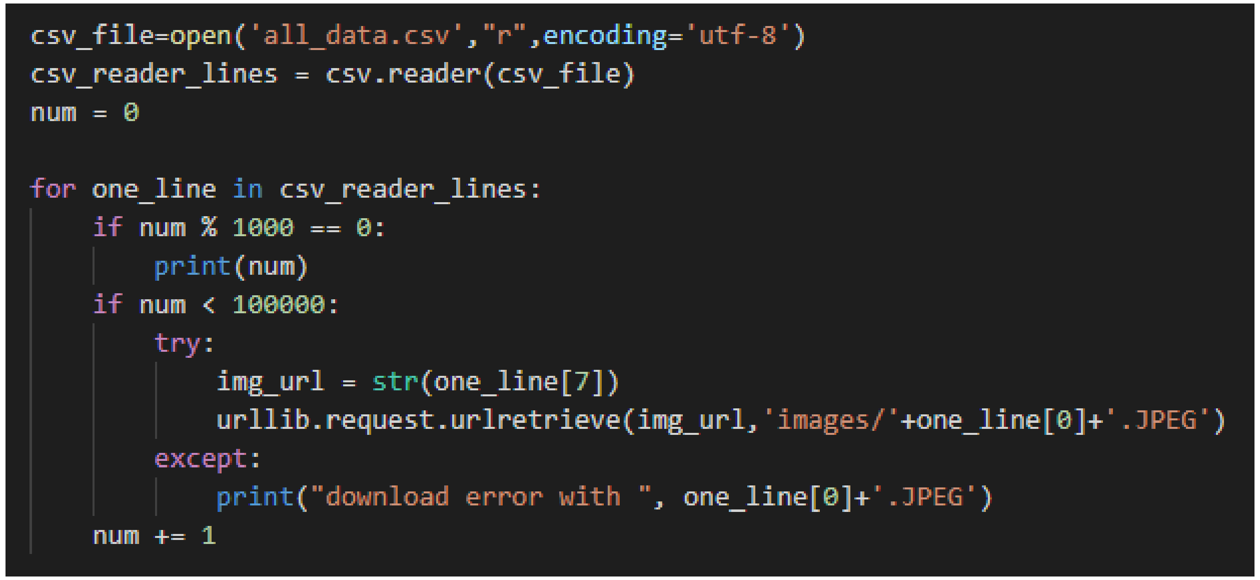


1	id	created_at	rating	score	sample_url	sample_width	sample_height	preview_url	tags
2	1	1264803292	s	37	http://safebooru.org/sampl	850	638	http://safebooru.org/thumbnails/1/thumb	1girl bag black_hair blush bob_cut bowieknife breath
3	2	1264803292	s	12	http://safebooru.org/sampl	850	1208	http://safebooru.org/thumbnails/1/thumb	barding black cape celty_sturluson dress dullahan dur
4	3	1264803298	s	8	http://safebooru.org/sampl	850	599	http://safebooru.org/thumbnails/1/thumb	blue_eyes blush brown_hair original scan takoyaki_(ro
5	4	1264803299	s	5	http://safebooru.org/sampl	850	519	http://safebooru.org/thumbnails/1/thumb	game_cg hagall_valkyr mecha_musume shirogane_nc
6	6	1264803304	s	11	http://safebooru.org/sampl	850	601	http://safebooru.org/thumbnails/1/thumb	blush idolmaster kisaragi_chihaya komi_zumiko pand
7	7	1264803305	s	2	http://safebooru.org/sampl	850	531	http://safebooru.org/thumbnails/1/thumb	blonde_hair detached_sleeves gloves green_eyes hair
8	8	1264803306	s	3	http://safebooru.org/sampl	850	638	http://safebooru.org/thumbnails/1/thumb	absolute_terror_field clouds dust electricity eva-02 lig
9	9	1264803306	s	4	http://safebooru.org/sampl	850	378	http://safebooru.org/thumbnails/1/thumb	armor cleavage game_cg hagall_valkyr shirogane_no
10	10	1264803308	s	7	http://safebooru.org/sampl	850	567	http://safebooru.org/thumbnails/1/thumb	bdsm bed blonde_hair bondage bow broken broken
11	11	1264803309	s	7	http://safebooru.org/sampl	850	601	http://safebooru.org/thumbnails/1/thumb	2_sing_4_u_(vocaloid) alternate_costume black_dress
12	12	1264803309	s	8	http://safebooru.org/image	1600	1200	http://safebooru.org/thumbnails/1/thumb	bandaid game cg kurushima shiho seifuku shirogane

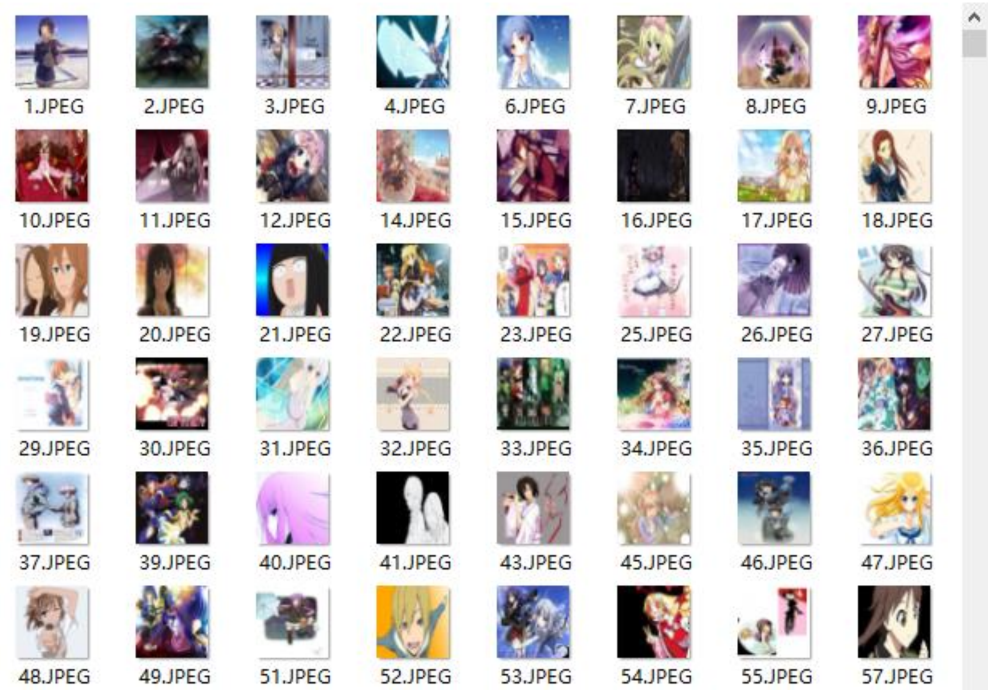
- tag: dividing image into different groups
- preview_url: downloading small size image

Data Pre-process 1

Downloading images using “download.py”.

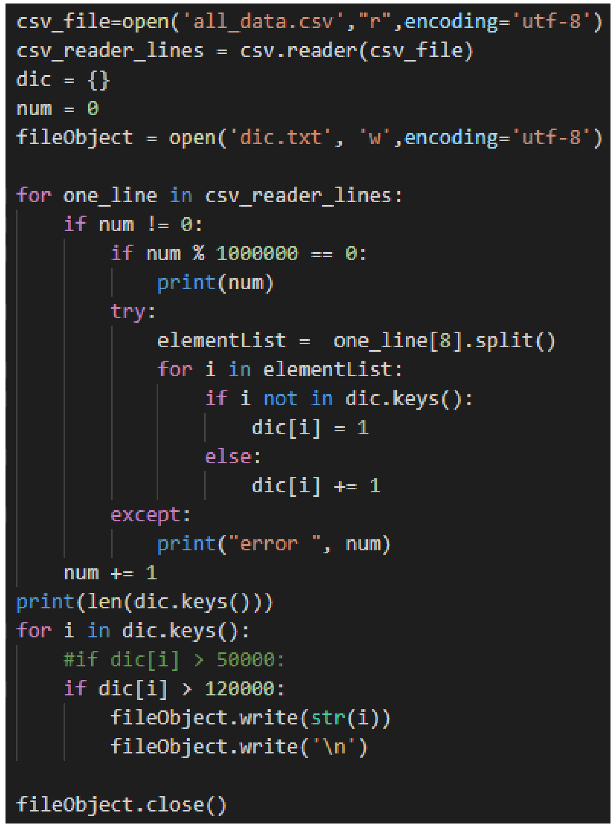


```python
csv_file=open('all_data.csv',"r",encoding='utf-8')
csv_reader_lines = csv.reader(csv_file)
num = 0

for one_line in csv_reader_lines:
    if num % 1000 == 0:
        print(num)
    if num < 100000:
        try:
            img_url = str(one_line[7])
            urllib.request.urlretrieve(img_url,'images/'+one_line[0]+'.JPEG')
        except:
            print("download error with ", one_line[0]+'.JPEG')
    num += 1
```

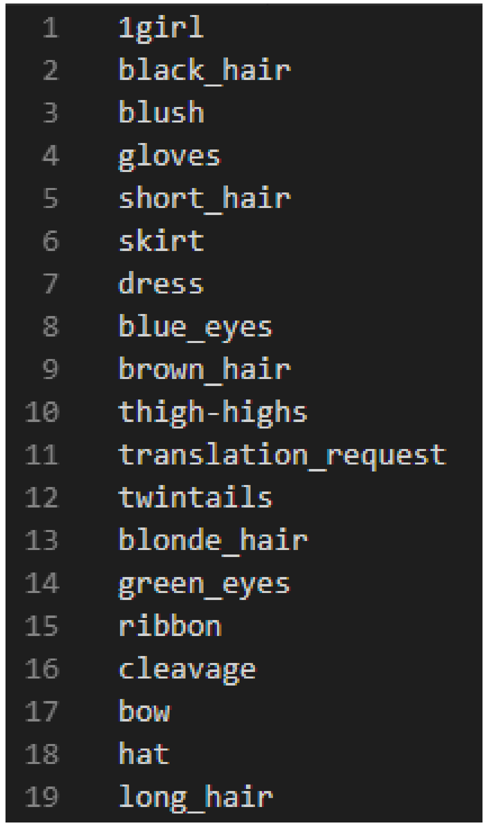


Data Pre-process 2

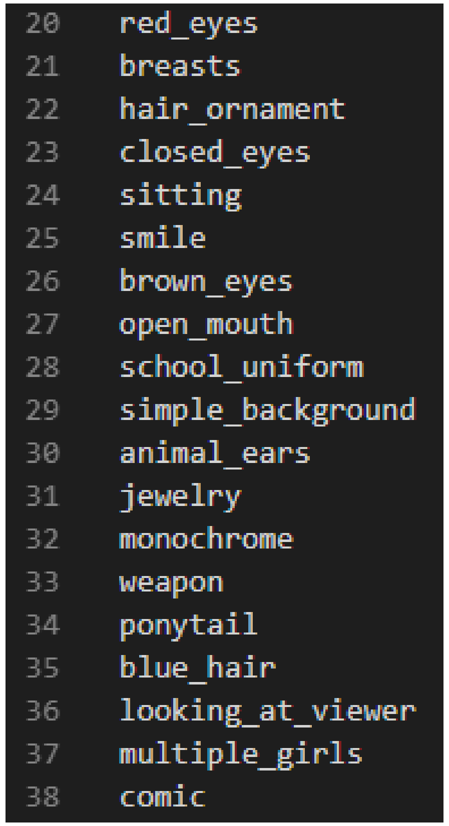
Generating group name dictionary using “set_dic.py”.



```python
csv_file=open('all_data.csv',"r",encoding='utf-8')
csv_reader_lines = csv.reader(csv_file)
dic = {}
num = 0
fileObject = open('dic.txt', 'w',encoding='utf-8')

for one_line in csv_reader_lines:
    if num != 0:
        if num % 1000000 == 0:
            print(num)
        try:
            elementList =  one_line[8].split()
            for i in elementList:
                if i not in dic.keys():
                    dic[i] = 1
                else:
                    dic[i] += 1
        except:
            print("error ", num)
    num += 1
print(len(dic.keys()))
for i in dic.keys():
    #if dic[i] > 50000:
    if dic[i] > 120000:
        fileObject.write(str(i))
        fileObject.write('\n')

fileObject.close()
```



```
1girl
black_hair
blush
gloves
short_hair
skirt
dress
blue_eyes
brown_hair
thigh-highs
translation_request
twintails
blonde_hair
green_eyes
ribbon
cleavage
bow
hat
long_hair
```



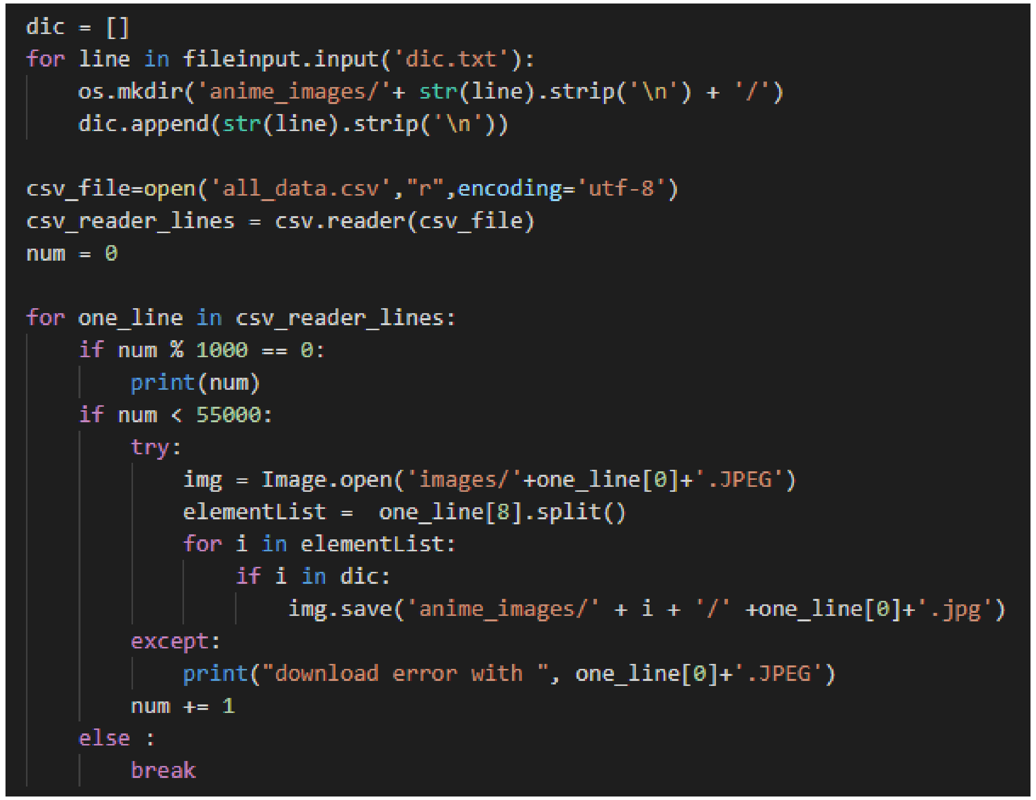
```
red_eyes
breasts
hair_ornament
closed_eyes
sitting
smile
brown_eyes
open_mouth
school_uniform
simple_background
animal_ears
jewelry
monochrome
weapon
ponytail
blue_hair
looking_at_viewer
multiple_girls
comic
```

38 groups in total.

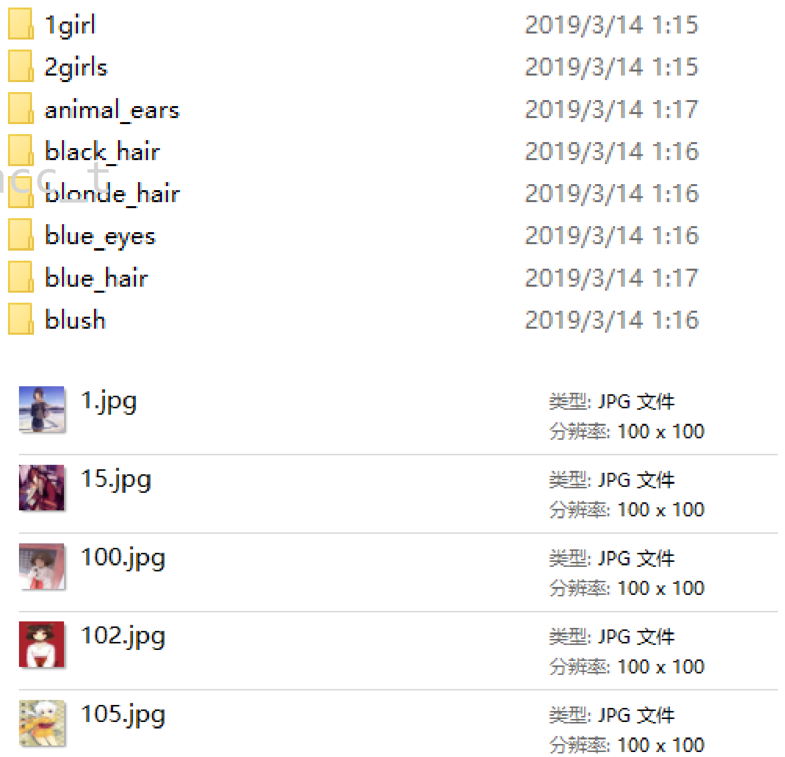
Saved in “dic.txt”

Data Pre-process 3

Resizing and grouping image using “deal.py”



```python
dic = []
for line in fileinput.input('dic.txt'):
    os.mkdir('anime_images/'+ str(line).strip('\n') + '/')
    dic.append(str(line).strip('\n'))

csv_file=open('all_data.csv',"r",encoding='utf-8')
csv_reader_lines = csv.reader(csv_file)
num = 0

for one_line in csv_reader_lines:
    if num % 1000 == 0:
        print(num)
    if num < 55000:
        try:
            img = Image.open('images/'+one_line[0]+'.JPEG')
            elementList =  one_line[8].split()
            for i in elementList:
                if i in dic:
                    img.save('anime_images/' + i + '/' +one_line[0]+'.jpg')
        except:
            print("download error with ", one_line[0]+'.JPEG')
        num += 1
    else :
        break
```

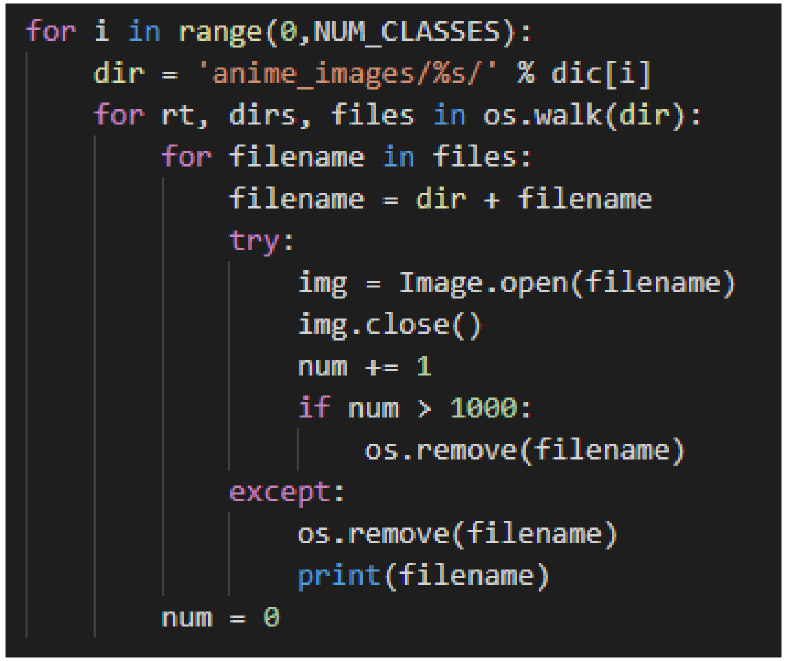


```
1girl          2019/3/14 1:15
2girls         2019/3/14 1:15
animal_ears    2019/3/14 1:17
black_hair     2019/3/14 1:16
blonde_hair    2019/3/14 1:16
blue_eyes      2019/3/14 1:16
blue_hair      2019/3/14 1:17
blush          2019/3/14 1:16

1.jpg          类型: JPG 文件
               分辨率: 100 x 100
15.jpg         类型: JPG 文件
               分辨率: 100 x 100
100.jpg        类型: JPG 文件
               分辨率: 100 x 100
102.jpg        类型: JPG 文件
               分辨率: 100 x 100
105.jpg        类型: JPG 文件
               分辨率: 100 x 100
```

Resizing to 100 x 100

Data Pre-process 4

- Checking the validation of image using “check.py”. (Significant)

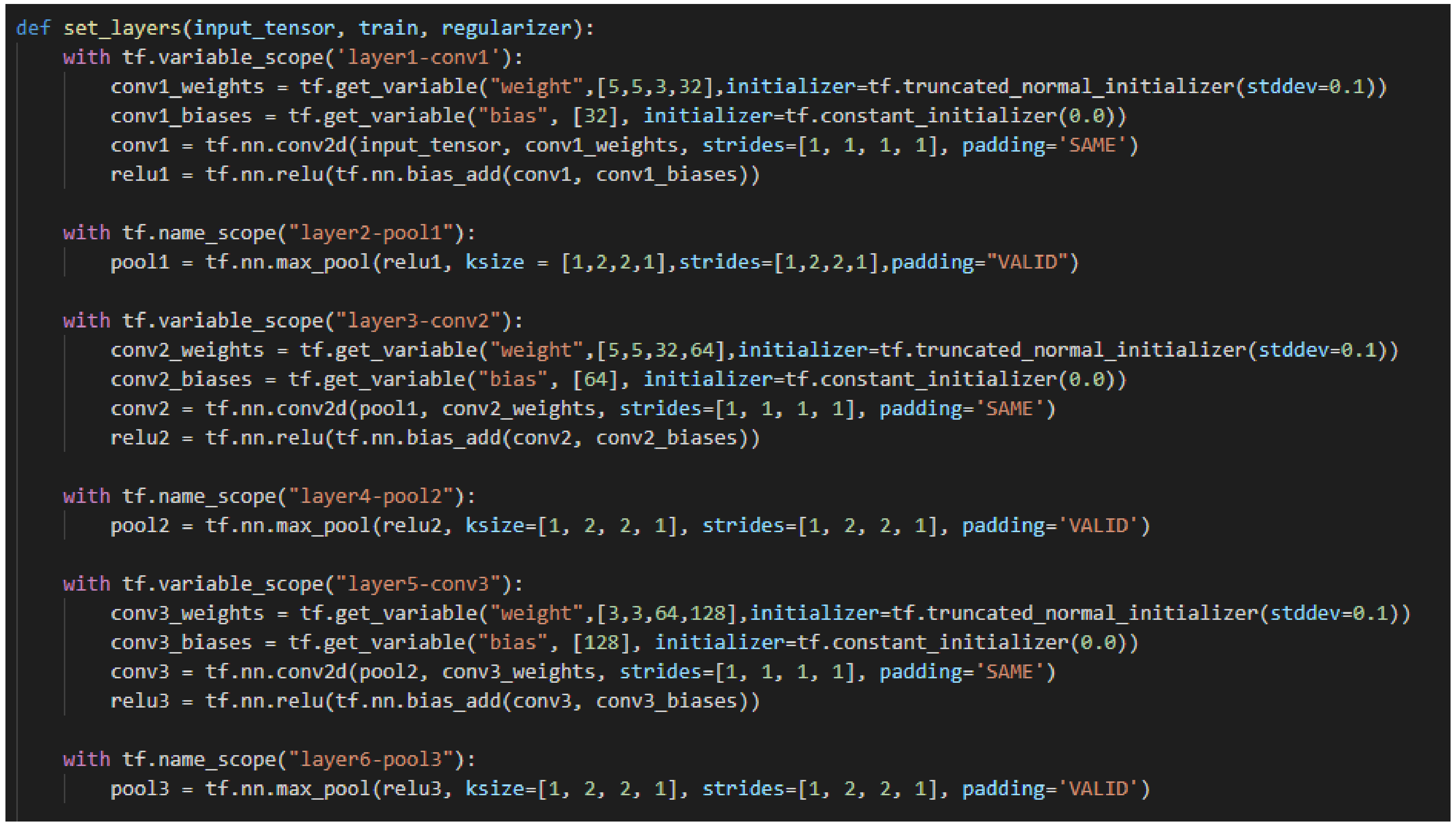


```
for i in range(0,NUM_CLASSES):
    dir = 'anime_images/%s/' % dic[i]
    for rt, dirs, files in os.walk(dir):
        for filename in files:
            filename = dir + filename
            try:
                img = Image.open(filename)
                img.close()
                num += 1
                if num > 1000:
                    os.remove(filename)
            except:
                os.remove(filename)
                print(filename)
        num = 0
```

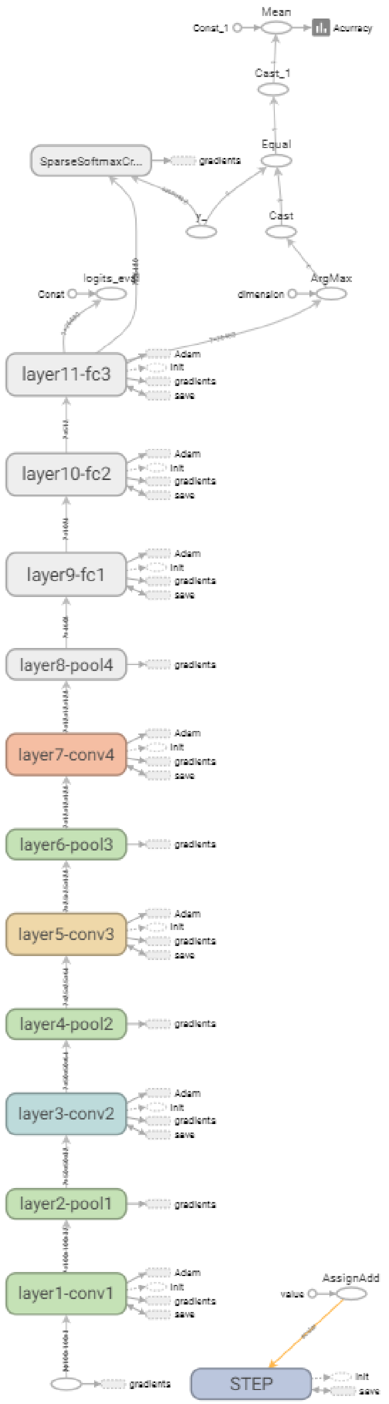
- Training process will break if the input image is invalid.
- This function set 1000 as the maximum number of each group.

Training Model 1

- 11 Layers CNN

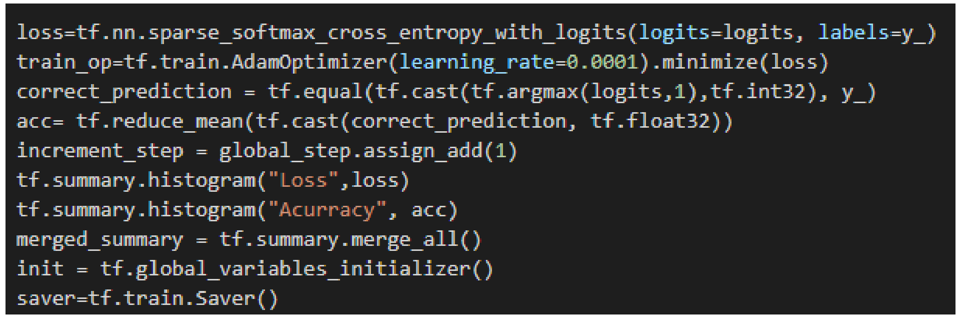


```
def set_layers(input_tensor, train, regularizer):
    with tf.variable_scope('layer1-conv1'):
        conv1_weights = tf.get_variable("weight",[5,5,3,32],initializer=tf.truncated_normal_initializer(stddev=0.1))
        conv1_biases = tf.get_variable("bias", [32], initializer=tf.constant_initializer(0.0))
        conv1 = tf.nn.conv2d(input_tensor, conv1_weights, strides=[1, 1, 1, 1], padding='SAME')
        relu1 = tf.nn.relu(tf.nn.bias_add(conv1, conv1_biases))

    with tf.name_scope("layer2-pool1"):
        pool1 = tf.nn.max_pool(relu1, ksize = [1,2,2,1],strides=[1,2,2,1],padding="VALID")

    with tf.variable_scope("layer3-conv2"):
        conv2_weights = tf.get_variable("weight",[5,5,32,64],initializer=tf.truncated_normal_initializer(stddev=0.1))
        conv2_biases = tf.get_variable("bias", [64], initializer=tf.constant_initializer(0.0))
        conv2 = tf.nn.conv2d(pool1, conv2_weights, strides=[1, 1, 1, 1], padding='SAME')
        relu2 = tf.nn.relu(tf.nn.bias_add(conv2, conv2_biases))

    with tf.name_scope("layer4-pool2"):
        pool2 = tf.nn.max_pool(relu2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')

    with tf.variable_scope("layer5-conv3"):
        conv3_weights = tf.get_variable("weight",[3,3,64,128],initializer=tf.truncated_normal_initializer(stddev=0.1))
        conv3_biases = tf.get_variable("bias", [128], initializer=tf.constant_initializer(0.0))
        conv3 = tf.nn.conv2d(pool2, conv3_weights, strides=[1, 1, 1, 1], padding='SAME')
        relu3 = tf.nn.relu(tf.nn.bias_add(conv3, conv3_biases))

    with tf.name_scope("layer6-pool3"):
        pool3 = tf.nn.max_pool(relu3, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')
```



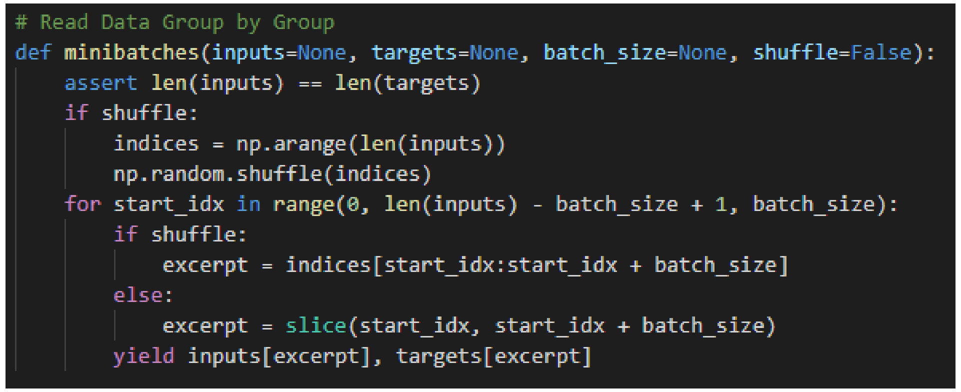
Training Model 2

- Calculating Accuracy, Loss and Prediction



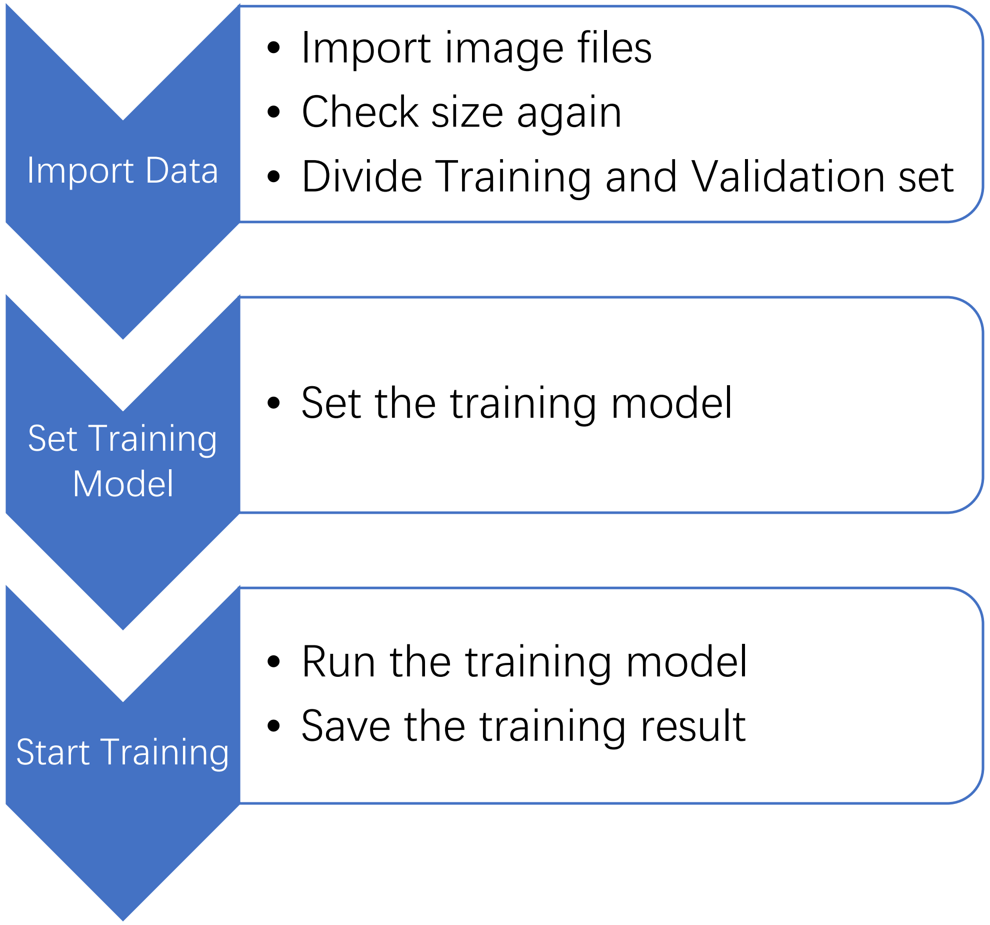
```
loss=tf.nn.sparse_softmax_cross_entropy_with_logits(logits=logits, labels=y_)
train_op=tf.train.AdamOptimizer(learning_rate=0.0001).minimize(loss)
correct_prediction = tf.equal(tf.cast(tf.argmax(logits,1),tf.int32), y_)
acc= tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
increment_step = global_step.assign_add(1)
tf.summary.histogram("Loss",loss)
tf.summary.histogram("Acurracy", acc)
merged_summary = tf.summary.merge_all()
init = tf.global_variables_initializer()
saver=tf.train.Saver()
```

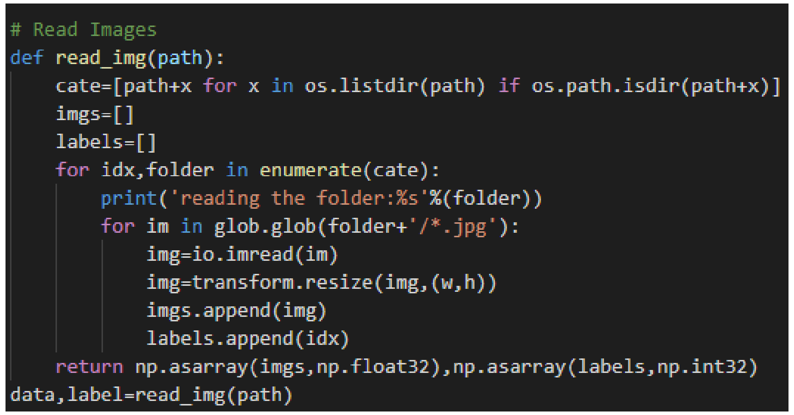
- Read Data Group by Group using “minibatches”



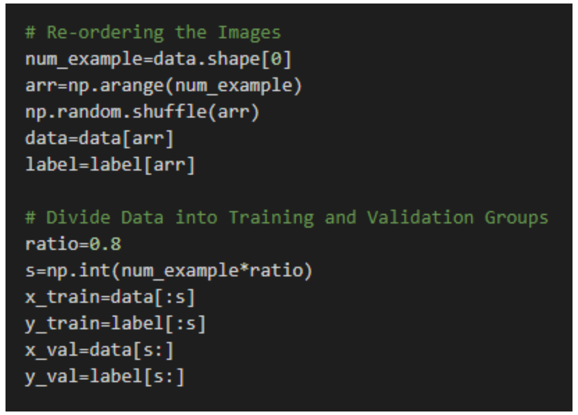
```
# Read Data Group by Group
def minibatches(inputs=None, targets=None, batch_size=None, shuffle=False):
    assert len(inputs) == len(targets)
    if shuffle:
        indices = np.arange(len(inputs))
        np.random.shuffle(indices)
    for start_idx in range(0, len(inputs) - batch_size + 1, batch_size):
        if shuffle:
            excerpt = indices[start_idx:start_idx + batch_size]
        else:
            excerpt = slice(start_idx, start_idx + batch_size)
        yield inputs[excerpt], targets[excerpt]
```

Training Process

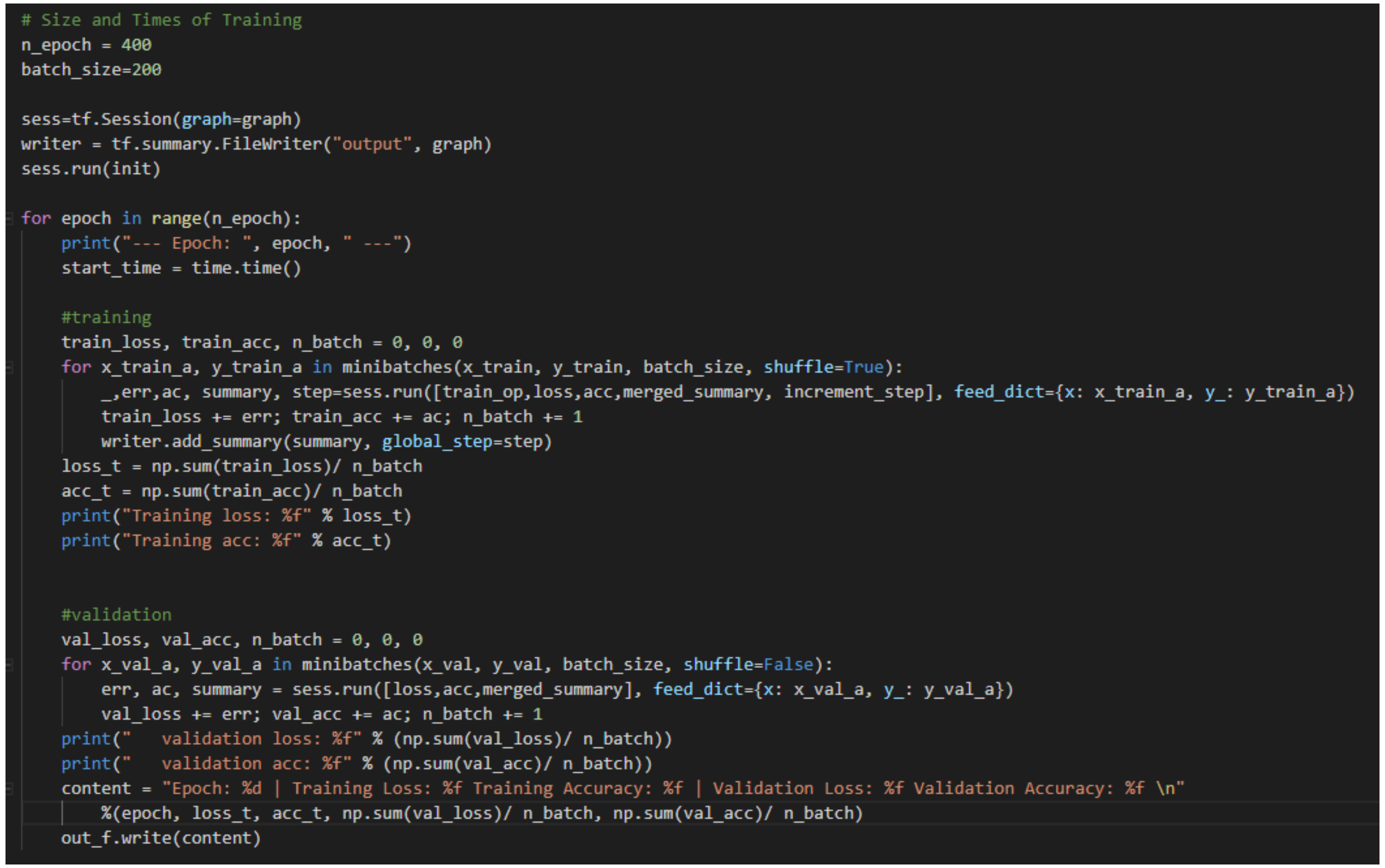




```
# Read Images
def read_img(path):
    cate=[path+x for x in os.listdir(path) if os.path.isdir(path+x)]
    imgs=[]
    labels=[]
    for idx,folder in enumerate(cate):
        print('reading the folder:%s'%(folder))
        for im in glob.glob(folder+'/*.jpg'):
            img=io.imread(im)
            img=transform.resize(img,(w,h))
            imgs.append(img)
            labels.append(idx)
    return np.asarray(imgs,np.float32),np.asarray(labels,np.int32)
data,label=read_img(path)
```



```
# Re-ordering the Images
num_example=data.shape[0]
arr=np.arange(num_example)
np.random.shuffle(arr)
data=data[arr]
label=label[arr]

# Divide Data into Training and Validation Groups
ratio=0.8
s=np.int(num_example*ratio)
x_train=data[:s]
y_train=label[:s]
x_val=data[s:]
y_val=label[s:]
```

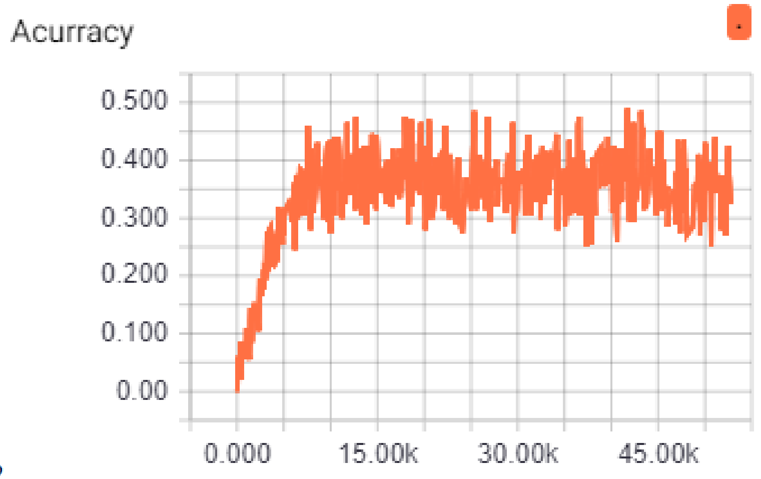


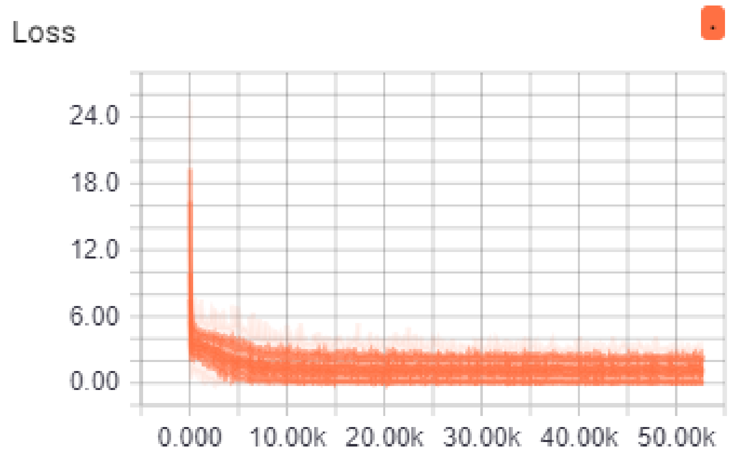
```
# Size and Times of Training
n_epoch = 400
batch_size=200

sess=tf.Session(graph=graph)
writer = tf.summary.FileWriter("output", graph)
sess.run(init)

for epoch in range(n_epoch):
    print("--- Epoch: ", epoch, " ---")
    start_time = time.time()

    #training
    train_loss, train_acc, n_batch = 0, 0, 0
    for x_train_a, y_train_a in minibatches(x_train, y_train, batch_size, shuffle=True):
        _,err,ac, summary, step=sess.run([train_op,loss,acc,merged_summary, increment_step], feed_dict={x: x_train_a, y_: y_train_a})
        train_loss += err; train_acc += ac; n_batch += 1
        writer.add_summary(summary, global_step=step)
    loss_t = np.sum(train_loss)/ n_batch
    acc_t = np.sum(train_acc)/ n_batch
    print("Training loss: %f" % loss_t)
    print("Training acc: %f" % acc_t)


    #validation
    val_loss, val_acc, n_batch = 0, 0, 0
    for x_val_a, y_val_a in minibatches(x_val, y_val, batch_size, shuffle=False):
        err, ac, summary = sess.run([loss,acc,merged_summary], feed_dict={x: x_val_a, y_: y_val_a})
        val_loss += err; val_acc += ac; n_batch += 1
    print("   validation loss: %f" % (np.sum(val_loss)/ n_batch))
    print("   validation acc: %f" % (np.sum(val_acc)/ n_batch))
    content = "Epoch: %d | Training Loss: %f Training Accuracy: %f | Validation Loss: %f Validation Accuracy: %f \n"
        %(epoch, loss_t, acc_t, np.sum(val_loss)/ n_batch, np.sum(val_acc)/ n_batch)
    out_f.write(content)
```

Training Result

- 38 groups, 1000 images per group (max)
- rate: 0.0001, epoch: 400, batch size: 200
- maximum accuracy: 0.48
- maximum avg accuracy: 0.37

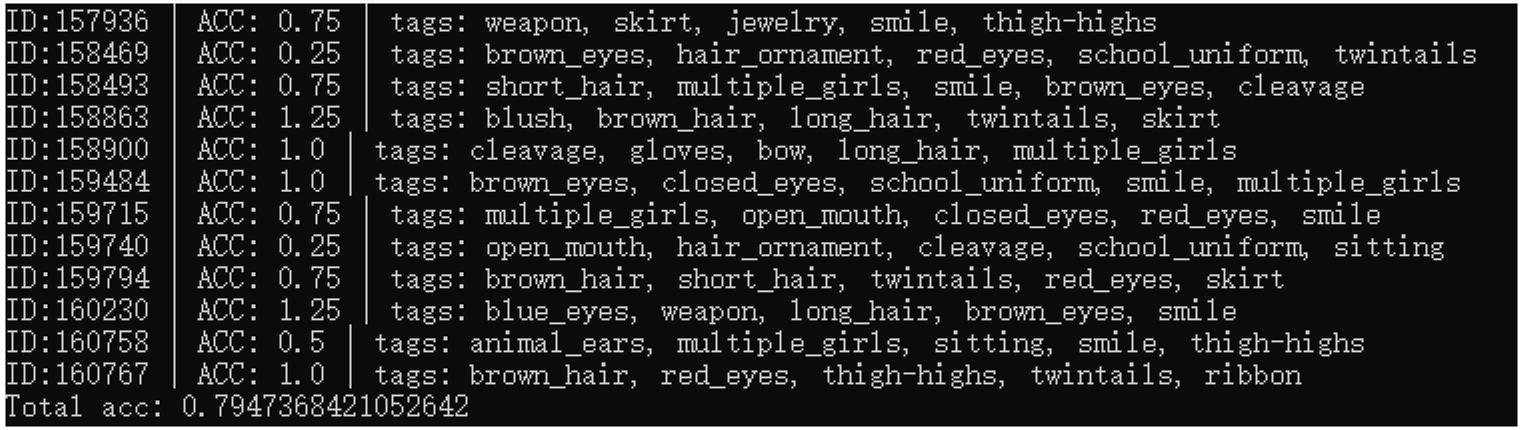
```
Epoch: 10 | Training Loss: 685.482465 Training Accuracy: 0.091778 | Validation Loss: 747.077000 Validation Accuracy: 0.050152
Epoch: 11 | Training Loss: 675.727199 Training Accuracy: 0.100852 | Validation Loss: 745.893584 Validation Accuracy: 0.052727
Epoch: 12 | Training Loss: 667.745602 Training Accuracy: 0.109185 | Validation Loss: 746.314986 Validation Accuracy: 0.056212
Epoch: 13 | Training Loss: 658.837037 Training Accuracy: 0.116037 | Validation Loss: 749.371804 Validation Accuracy: 0.057424
Epoch: 14 | Training Loss: 650.143403 Training Accuracy: 0.123370 | Validation Loss: 750.615826 Validation Accuracy: 0.055455
Epoch: 15 | Training Loss: 642.293056 Training Accuracy: 0.128704 | Validation Loss: 756.854226 Validation Accuracy: 0.054242
Epoch: 16 | Training Loss: 633.080208 Training Accuracy: 0.138222 | Validation Loss: 758.538767 Validation Accuracy: 0.056667
Epoch: 17 | Training Loss: 625.214120 Training Accuracy: 0.145370 | Validation Loss: 761.877782 Validation Accuracy: 0.050455
Epoch: 18 | Training Loss: 615.262500 Training Accuracy: 0.156185 | Validation Loss: 766.969460 Validation Accuracy: 0.053788
Epoch: 19 | Training Loss: 606.018461 Training Accuracy: 0.160704 | Validation Loss: 778.308239 Validation Accuracy: 0.053939
Epoch: 20 | Training Loss: 597.363079 Training Accuracy: 0.169556 | Validation Loss: 781.184659 Validation Accuracy: 0.053182
Epoch: 21 | Training Loss: 587.301100 Training Accuracy: 0.179074 | Validation Loss: 789.371330 Validation Accuracy: 0.053939
Epoch: 22 | Training Loss: 577.464352 Training Accuracy: 0.187407 | Validation Loss: 790.603693 Validation Accuracy: 0.047273
Epoch: 23 | Training Loss: 568.286458 Training Accuracy: 0.196185 | Validation Loss: 801.736861 Validation Accuracy: 0.050455
Epoch: 24 | Training Loss: 559.532986 Training Accuracy: 0.200593 | Validation Loss: 809.497041 Validation Accuracy: 0.049242
Epoch: 25 | Training Loss: 549.562442 Training Accuracy: 0.211407 | Validation Loss: 814.218632 Validation Accuracy: 0.051364
```





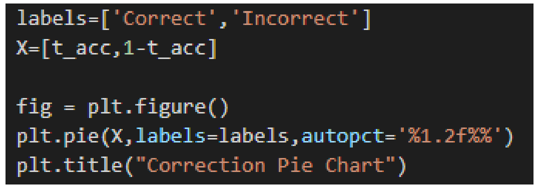
Test 1

- Pictures in raw data but out of training set
- Share more than 15 common tags compared with training tags
- Pick 6 highest prediction, each correct prediction provide 0.25 acc
- 95 images meet the criteria

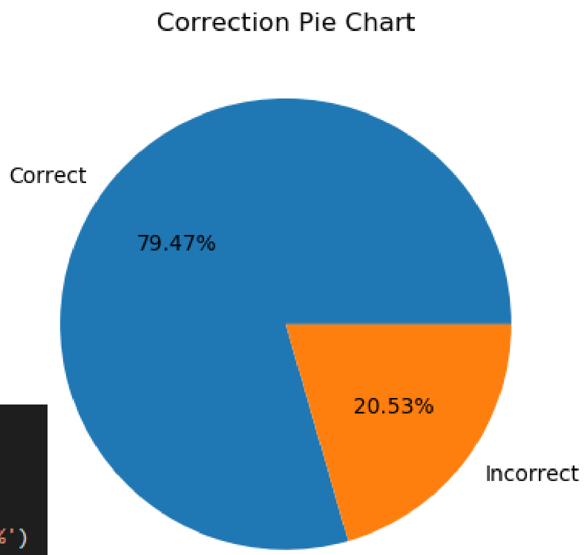


```
ID:157936 | ACC: 0.75 | tags: weapon, skirt, jewelry, smile, thigh-highs
ID:158469 | ACC: 0.25 | tags: brown_eyes, hair_ornament, red_eyes, school_uniform, twintails
ID:158493 | ACC: 0.75 | tags: short_hair, multiple_girls, smile, brown_eyes, cleavage
ID:158863 | ACC: 1.25 | tags: blush, brown_hair, long_hair, twintails, skirt
ID:158900 | ACC: 1.0 | tags: cleavage, gloves, bow, long_hair, multiple_girls
ID:159484 | ACC: 1.0 | tags: brown_eyes, closed_eyes, school_uniform, smile, multiple_girls
ID:159715 | ACC: 0.75 | tags: multiple_girls, open_mouth, closed_eyes, red_eyes, smile
ID:159740 | ACC: 0.25 | tags: open_mouth, hair_ornament, cleavage, school_uniform, sitting
ID:159794 | ACC: 0.75 | tags: brown_hair, short_hair, twintails, red_eyes, skirt
ID:160230 | ACC: 1.25 | tags: blue_eyes, weapon, long_hair, brown_eyes, smile
ID:160758 | ACC: 0.5 | tags: animal_ears, multiple_girls, sitting, smile, thigh-highs
ID:160767 | ACC: 1.0 | tags: brown_hair, red_eyes, thigh-highs, twintails, ribbon
Total acc: 0.7947368421052642
```

Accuracy: 0.7947

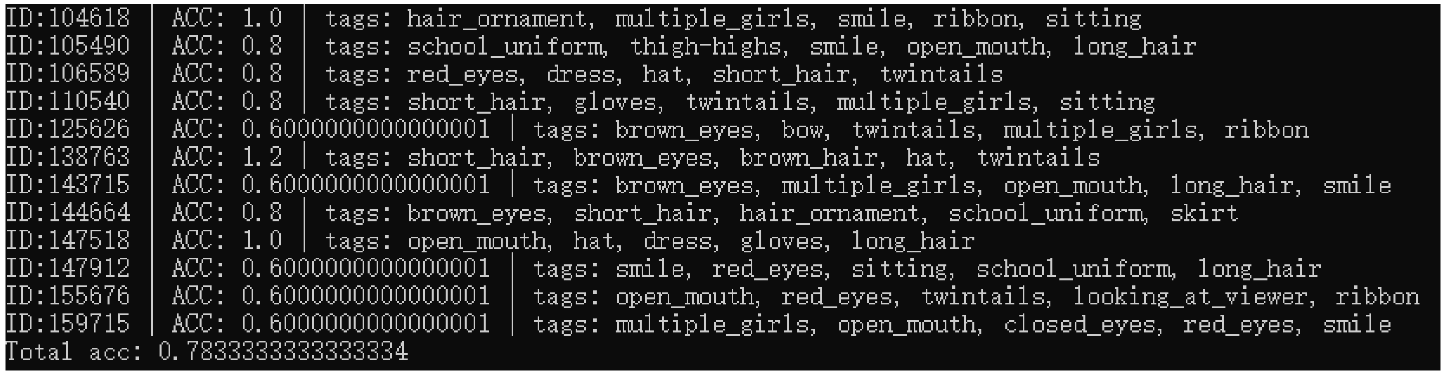


```
labels=['Correct','Incorrect']
X=[t_acc,1-t_acc]

fig = plt.figure()
plt.pie(X,labels=labels,autopct='%1.2f%%')
plt.title("Correction Pie Chart")
```



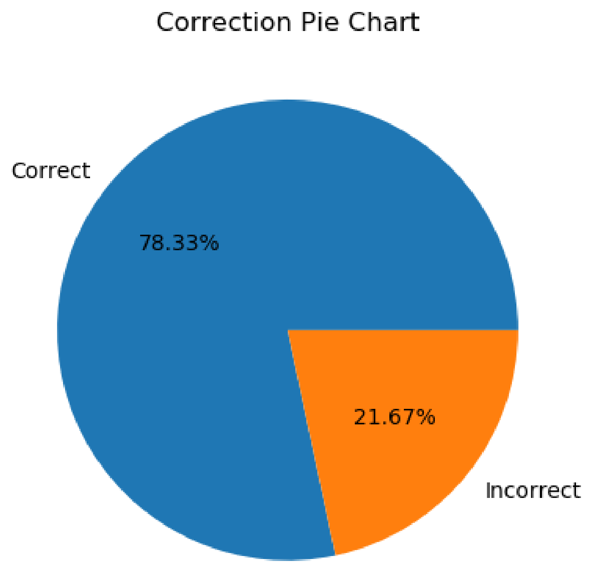
Test 2

- Pictures in raw data but out of training set
- Share more than 20 common tags compared with training tags
- Pick 6 highest prediction, each correct prediction provide 0.2 acc
- 12 images meet the criteria



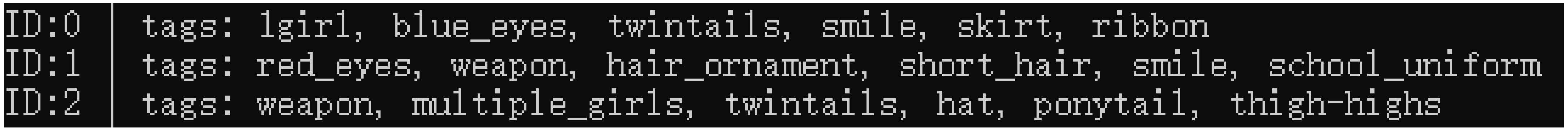
```
ID:104618 | ACC: 1.0 | tags: hair_ornament, multiple_girls, smile, ribbon, sitting
ID:105490 | ACC: 0.8 | tags: school_uniform, thigh-highs, smile, open_mouth, long_hair
ID:106589 | ACC: 0.8 | tags: red_eyes, dress, hat, short_hair, twintails
ID:110540 | ACC: 0.8 | tags: short_hair, gloves, twintails, multiple_girls, sitting
ID:125626 | ACC: 0.6000000000000001 | tags: brown_eyes, bow, twintails, multiple_girls, ribbon
ID:138763 | ACC: 1.2 | tags: short_hair, brown_eyes, brown_hair, hat, twintails
ID:143715 | ACC: 0.6000000000000001 | tags: brown_eyes, multiple_girls, open_mouth, long_hair, smile
ID:144664 | ACC: 0.8 | tags: brown_eyes, short_hair, hair_ornament, school_uniform, skirt
ID:147518 | ACC: 1.0 | tags: open_mouth, hat, dress, gloves, long_hair
ID:147912 | ACC: 0.6000000000000001 | tags: smile, red_eyes, sitting, school_uniform, long_hair
ID:155676 | ACC: 0.6000000000000001 | tags: open_mouth, red_eyes, twintails, looking_at_viewer, ribbon
ID:159715 | ACC: 0.6000000000000001 | tags: multiple_girls, open_mouth, closed_eyes, red_eyes, smile
Total acc: 0.7833333333333334
```

Accuracy: 0.7833

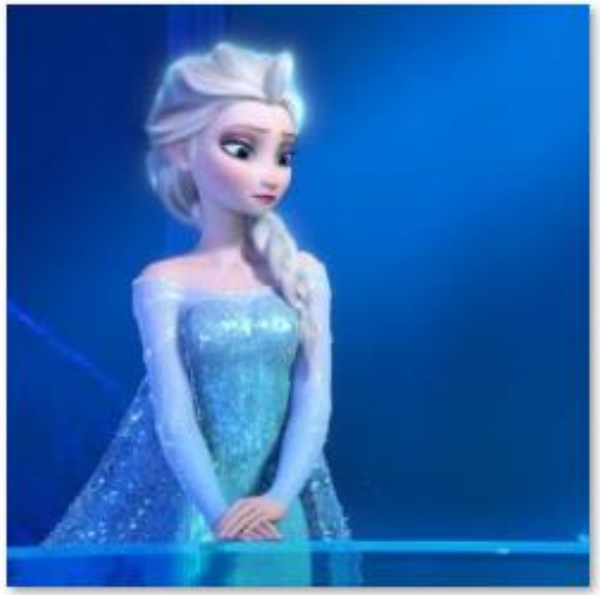


Test 3

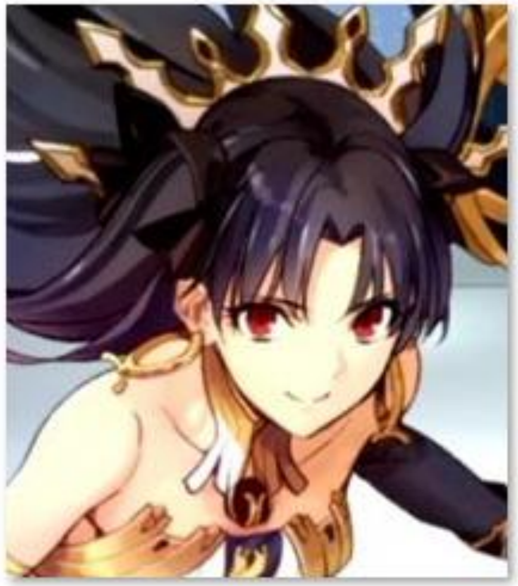
- Some tests with other screenshot out of database



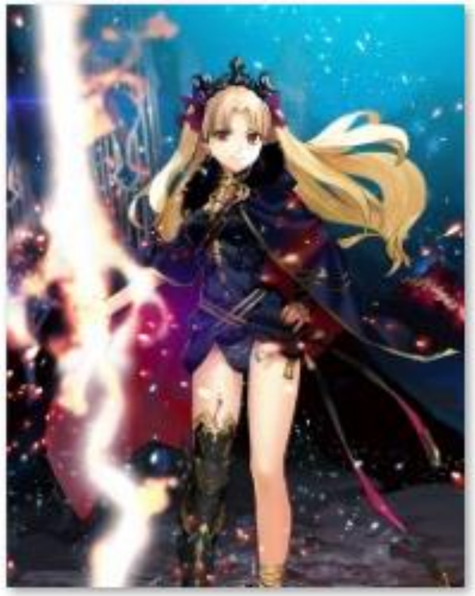
```
ID:0 | tags: 1girl, blue_eyes, twintails, smile, skirt, ribbon
ID:1 | tags: red_eyes, weapon, hair_ornament, short_hair, smile, school_uniform
ID:2 | tags: weapon, multiple_girls, twintails, hat, ponytail, thigh-highs
```



0.jpg



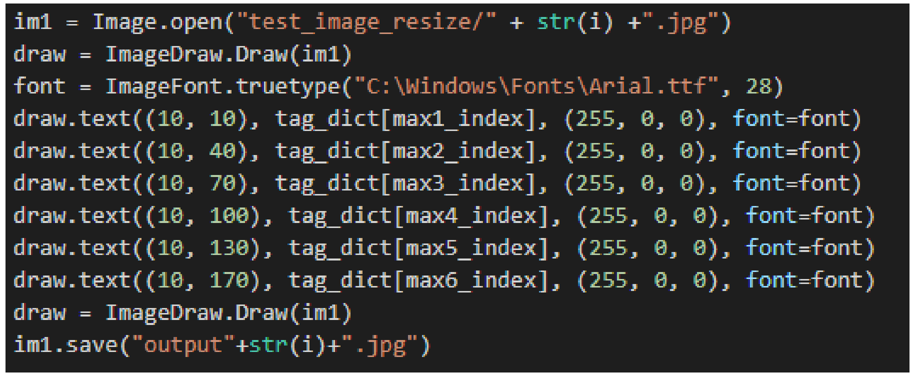
1.jpg



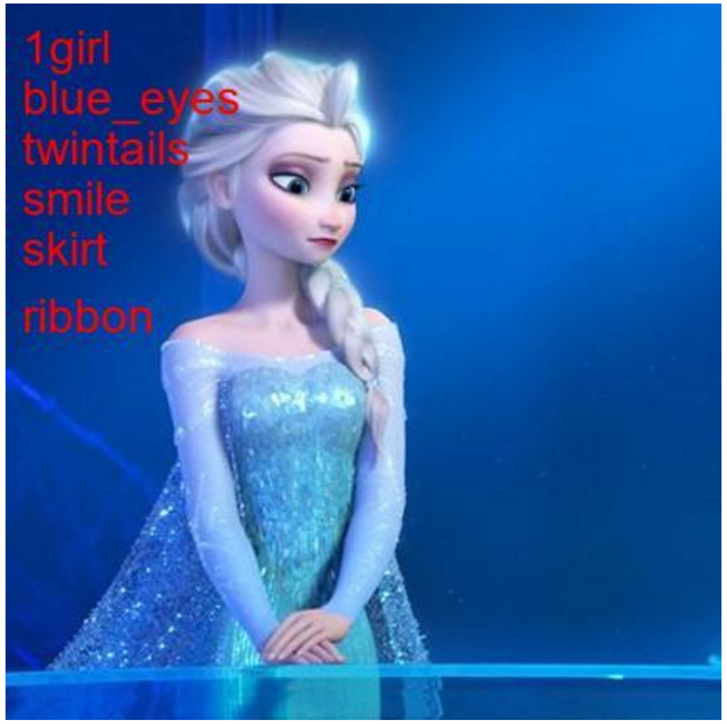
2.jpg

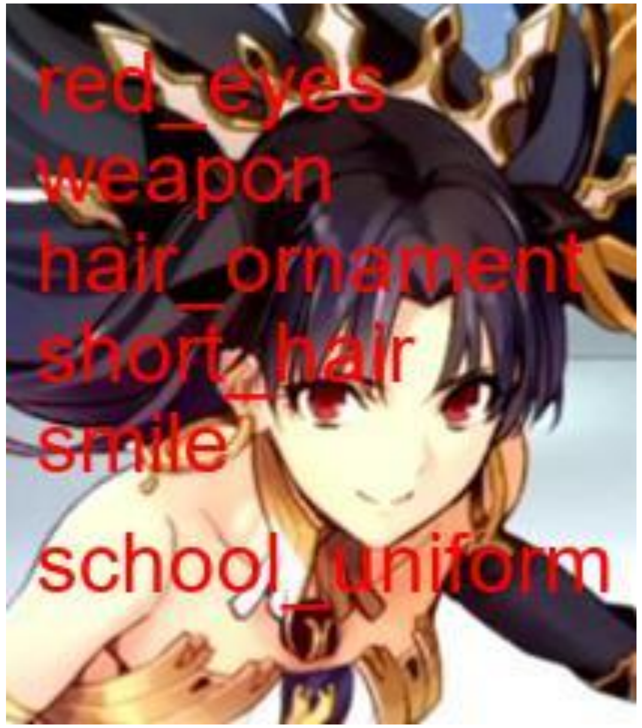
Advanced

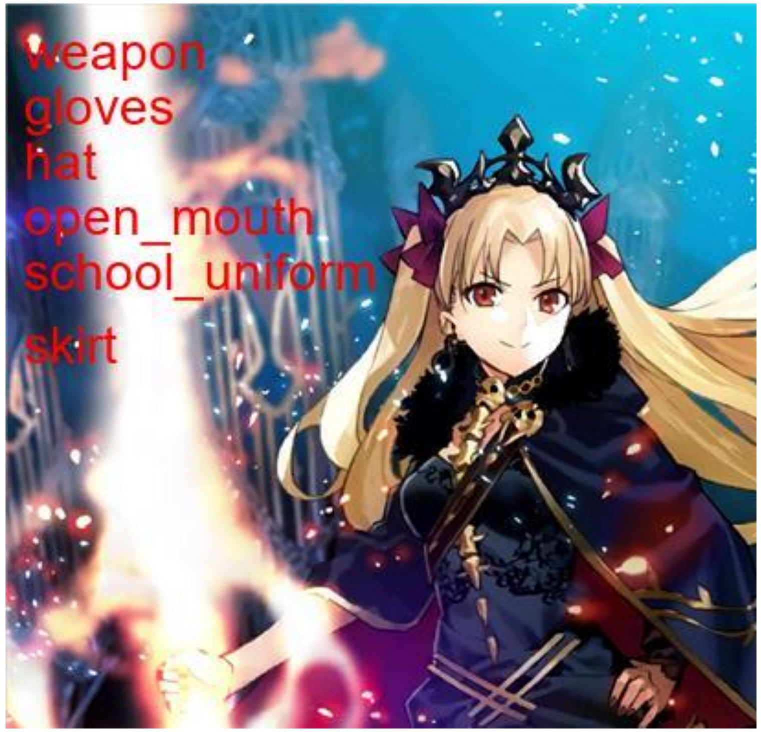
- Add tag to the image using “Draw”



```
im1 = Image.open("test_image_resize/" + str(i) +".jpg")
draw = ImageDraw.Draw(im1)
font = ImageFont.truetype("C:\Windows\Fonts\Arial.ttf", 28)
draw.text((10, 10), tag_dict[max1_index], (255, 0, 0), font=font)
draw.text((10, 40), tag_dict[max2_index], (255, 0, 0), font=font)
draw.text((10, 70), tag_dict[max3_index], (255, 0, 0), font=font)
draw.text((10, 100), tag_dict[max4_index], (255, 0, 0), font=font)
draw.text((10, 130), tag_dict[max5_index], (255, 0, 0), font=font)
draw.text((10, 170), tag_dict[max6_index], (255, 0, 0), font=font)
draw = ImageDraw.Draw(im1)
im1.save("output"+str(i)+".jpg")
```


1girl
blue_eyes
twintails
smile
skirt
ribbon


red_eyes
weapon
hair_ornament
short_hair
smile
school_uniform


weapon
gloves
hat
open_mouth
school_uniform
skirt