In [1]:
```python
# 1. Classification Using Hand-Crafted Features
# (a)
# Load VizWiz dataset
import os
import json
import requests
from pprint import PrettyPrinter

base_url = 'https://ivc.ischool.utexas.edu/VizWiz/data'
img_dir = '%s/Images/' %base_url
print(img_dir)

train_split = 'train'
train_file = '%s/Annotations/%s.json' %(base_url, train_split)
train_data = requests.get(train_file, allow_redirects=True)
print(train_file)

test_split = 'test'
test_file = '%s/Annotations/%s.json' %(base_url, test_split)
test_data = requests.get(test_file, allow_redirects=True)
print(test_file)

val_split = 'val'
val_file = '%s/Annotations/%s.json' %(base_url, val_split)
val_data = requests.get(val_file, allow_redirects=True)
print(val_file)
```

```
https://ivc.ischool.utexas.edu/VizWiz/data/Images/
https://ivc.ischool.utexas.edu/VizWiz/data/Annotations/train.json
https://ivc.ischool.utexas.edu/VizWiz/data/Annotations/test.json
https://ivc.ischool.utexas.edu/VizWiz/data/Annotations/val.json
```

In [2]:
```python
# Read the local file
training_data = train_data.json()
testing_data = test_data.json()
validation_data = val_data.json()
print("Length of training data:", len(training_data))
print("Length of test data:", len(testing_data))
print("Length of validation data:", len(validation_data))

image_name_train = []
question_train = []
label_train = []

image_name_val = []
question_val = []
label_val = []

image_name_test = []
question_test = []
label_test = []

num_train_VQs = 20000
for vq in training_data[0:num_train_VQs]:
    image_name_train.append(vq['image'])
    question_train.append(vq['question'])
    label_train.append(vq['answerable'])

num_val_VQs = 8000
for vq in validation_data[0:num_val_VQs]:
    image_name_val.append(vq['image'])
    question_val.append(vq['question'])
    label_val.append(vq['answerable'])

num_test_VQs = 3173
for vq in testing_data[0:num_test_VQs]:
    image_name_test.append(vq['image'])
    question_test.append(vq['question'])
#     label_test.append(vq['answerable'])

import pandas as pd
image_name_train = pd.DataFrame(image_name_train, columns=['image'])
image_name_val = pd.DataFrame(image_name_val, columns=['image'])
image_name_test = pd.DataFrame(image_name_test, columns=['image'])
question_train = pd.DataFrame(question_train, columns=['question'])
question_val = pd.DataFrame(question_val, columns=['question'])
question_test = pd.DataFrame(question_test, columns=['question'])

X_train = pd.concat([image_name_train, question_train], axis=1)
y_train = pd.DataFrame(label_train, columns=['label'])
X_val = pd.concat([image_name_val, question_val], axis=1)
y_val = pd.DataFrame(label_val, columns=['label'])
X_test = pd.concat([image_name_test, question_test], axis=1)
# y_test = pd.DataFrame(label_test, columns='label')
```

```
Length of training data: 20000
Length of test data: 8000
Length of validation data: 3173
```

In [12]:

```python
# (b)
# Use Microsoft Azure API to extract image-based features
subscription_key_vision = '412bc41b5b5844febf4d7cd63510fb4f'
vision_base_url = 'https://westcentralus.api.cognitive.microsoft.com/vis
ion/v1.0'
vision_analyze_url = vision_base_url + '/analyze?'
from time import sleep

def analyze_image(image_url):
    # Microsoft API headers, params, etc
    headers = {'Ocp-Apim-Subscription-key': subscription_key_vision}
    params = {'visualfeatures': 'Description, Tags'}
    data = {'url': image_url}
    # send request, get API response
    try:
        response = requests.post(vision_analyze_url,headers = headers,pa
rams=params,json=data)
    except:
        sleep(10)
        response = requests.post(vision_analyze_url,headers = headers,pa
rams=params,json=data)
#     response = requests.post(vision_analyze_url, headers=headers, para
ms=params, json=data)
    if (response.status_code == 200):
        analysis = response.json()
    else:
        print("get image {} failed".format(image_url))
        analysis = {"description":{"tags":[]}}
    return analysis

def extract_features(data):
    return {
        'tags': data['description']['tags'],
#          'confidence': data['tags'][0]['confidence']
    }

image_feature = {}
def get_image_feature(X):

    for i in range(20000):
        image_url = img_dir + '%s' %(X['image'][i])
        data = extract_features(analyze_image(image_url))
        tag_i = []
        for item in data['tags']:
            tag_i.append(item)
        tag_i_join = ' '.join(tag_i)
#          image_feature.append(tag_i_join)
        image_feature[str(i)] = tag_i_join
        if (i%500==0):
            print('get number',str(i))

    return image_feature
image_feature = get_image_feature(X_train)
```

```
get number 12500
get number 13000
get number 13500
get number 14000
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000014307.jpg failed
get number 14500
get number 15000
get number 15500
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000015541.jpg failed
get number 16000
get number 16500
get number 17000
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000017089.jpg failed
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000017311.jpg failed
get number 17500
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000017821.jpg failed
get number 18000
get number 18500
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000018603.jpg failed
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000018777.jpg failed
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000018938.jpg failed
get number 19000
get number 19500
get image https://ivc.ischool.utexas.edu/VizWiz/data/Images/VizWiz_trai
n_000000019757.jpg failed
```

In [13]:

```python
# Write image feature to csv file

import csv

data = pd.DataFrame()
indexlist = []
featurelist = []
for index,feature in image_feature.items():
    indexlist.append(index)
    featurelist.append(feature)
data["id"] = indexlist
data["image_feature"] = featurelist
data.columns = ["id", "image_feature"]
data.head()
data.to_csv('image_feature_train.csv', index=False)
```

In [ ]:
```
# Extract text features using Microsoft Azure
from time import sleep
subscription_key_text = 'e25225c679e74f61a2ab61924b41a866'
text_analytics_base_url = 'https://centralus.api.cognitive.microsoft.co
m/text/analytics/v2.0/'
key_phrase_api_url = text_analytics_base_url + 'keyPhrases'
question_feature = {}
def get_question_feature(question_train):

    for i in range(20000):

        question_json = question_train['question'][i]
        documents = {'documents': [{'id': i, 'text': question_json}]}
        headers = {"Ocp-Apim-Subscription-Key": subscription_key_text}
        maxiter = 10

        try:
            response = requests.post(key_phrase_api_url,headers = header
s,json=documents)
        except:
            sleep(10)
            response = requests.post(key_phrase_api_url,headers = header
s,json=documents)
        if(response.status_code == 200):
            question_json = response.json()['documents']
            question = pd.DataFrame(question_json)['keyPhrases']
            question = question.tolist()[0]
            tag_i=[]
            for item in question:
                tag_i.append(item)
            question = ' '.join(tag_i)
            question_feature[str(i)] = question
        else:
            print("not get",str(i))
            question_feature[str(i)] = ""
        if (i%500==0):
            print('get number',str(i))

    return question_feature
question_feature = get_question_feature(X_train)
#print(question_feature)
```

In [ ]:
```
# Write key phrase to csv file
data = pd.DataFrame()
indexlist = []
keywordlist = []
for index,keyword in question_feature.items():
    indexlist.append(index)
    keywordlist.append(keyword)
data["id"] = indexlist
data["question_keyword"] = keywordlist
data.columns = ["id", "question_keyword"]
data.head()
data.to_csv('question_feature_train.csv', index=False)
```

```
In [ ]:
```