GR5291 A Study on CNN Classification: Detecting Skin Cancer

Group 17

Luyue, Chen, lc3363
Guoyong, Xu, gx2138
Hanyu, Gao, hg2490
Yanzhu, Chen, yc3511
Zhaochen, Li, zl2685
Xuanhong, Ye, xy2387
Lulu, Wang, lw2836
Kanyan, Chen, kc3207

1. Introduction

The diagnosis of pigmented skin lesions is a vital process for detecting skin cancer. Thus dermatoscopy is a widely used diagnostic technique that improves the diagnosis of benign and malignant pigmented skin lesions in comparison to examination with the unaided eye. Dermatoscopic images are also a suitable source to train artificial neural networks to diagnose pigmented skin lesions automatically. Therefore, our project focuses on the classification of different types of dermatoscopic images, and train a CNN model to automatically detect skin cancer. Moreover, we will design a simple app to make classification based on the CNN model we have trained.

2. Project objective

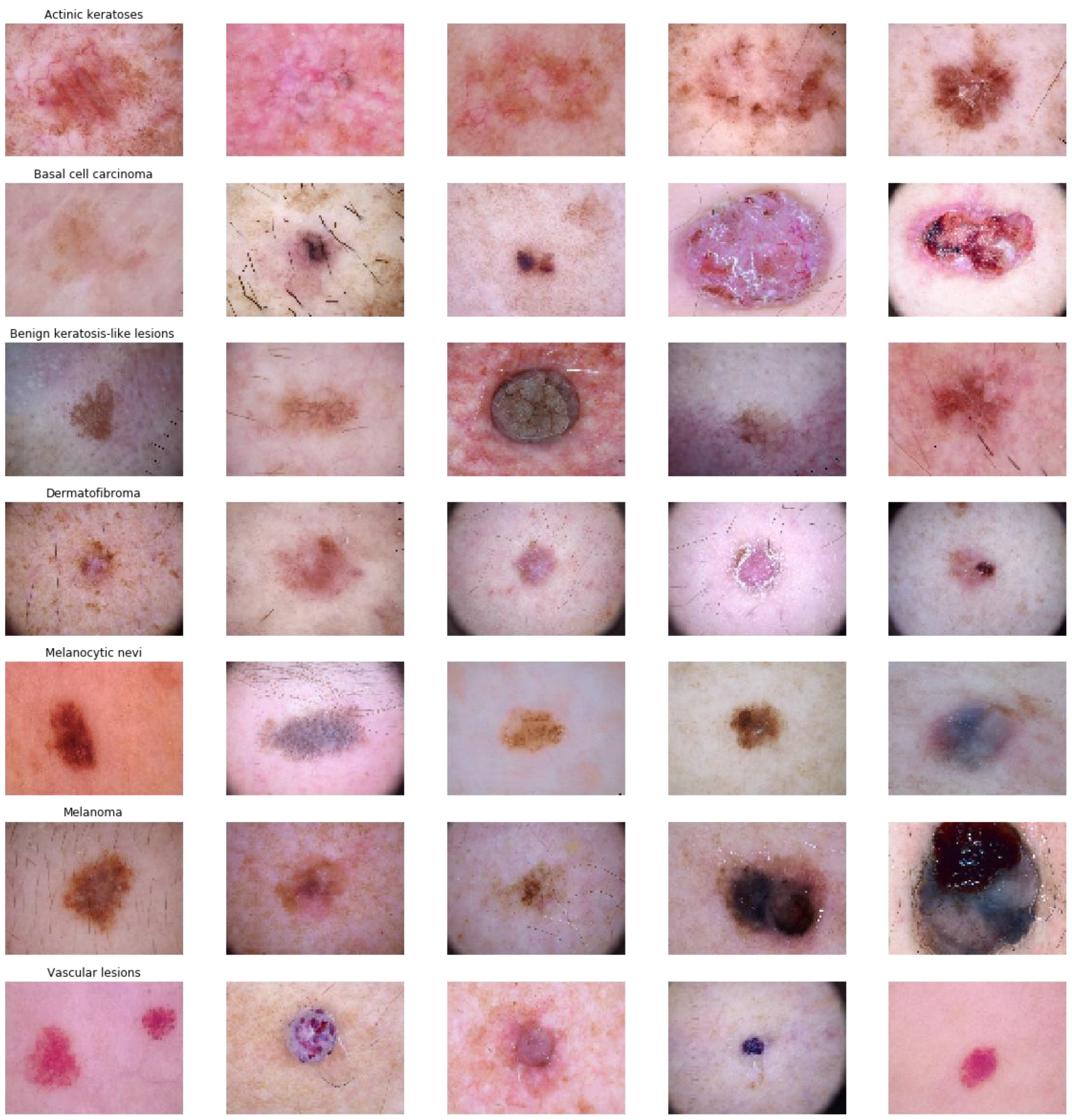
The target of our project is to train a model as accurate as possible to detect different kinds of skin cancer, given the dermatoscopic images. Another objective is to build a live web app to use our model.

3. Data source and description

Human Against Machine with 10000 training images (HAM10000) is a large dataset which contains 10015 dermatoscopic images and it also provides information on the performance of human expert diagnosis. The metadata includes leision_id, image_id, dx, dx_type, age, sex and localization. A sample is given below:

leision_id	image_id	dx	dx_type	age	sex	localization
HAM_0000118	ISIC_0027419	bkl	histo	80	male	scalp
HAM_0000118	ISIC_0025030	bkl	histo	80	male	scalp
HAM_0001466	ISIC_0031633	bkl	histo	75	male	ear
HAM_0002761	ISIC_0029068	bkl	histo	60	male	face
HAM_0005132	ISIC_0025837	bkl	histo	70	female	back

The image set of HAM10000 is a large collection of multi-source dermatoscopic images of common pigmented skin cancers. Some of the images are given below:



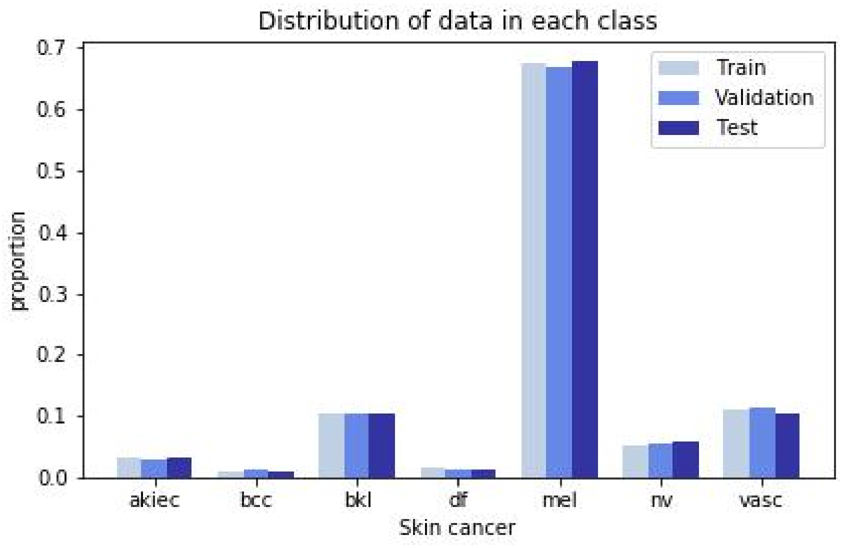
4. Planned methods of analysis

4.1 Exploratory data analysis

Histogram and box plot will be used in this part to reflect the relationship between different variables. To make the results more convincing, Kolmogorov-Smirnov test and Pearson's Chi-squared test will be conducted to support the conclusion.

4.2 CNN model

A fine tuned MobileNet CNN will be applied to classify skin lesions into seven classes. MobileNet's small size and speed makes it ideal for web deployment. Before the training process, the image data is labeled as akiec, bcc, bkl, df, mel, nv, vasc, denoting the seven types of pigmented skin lesions respectively, then it is randomly split into training, validation and testing set with the proportion equaling to 6:3:1 approximately. Further, each image is resized into dimensions of (224,224,3) and also rescaled from RGB vector into numbers between [0,1].



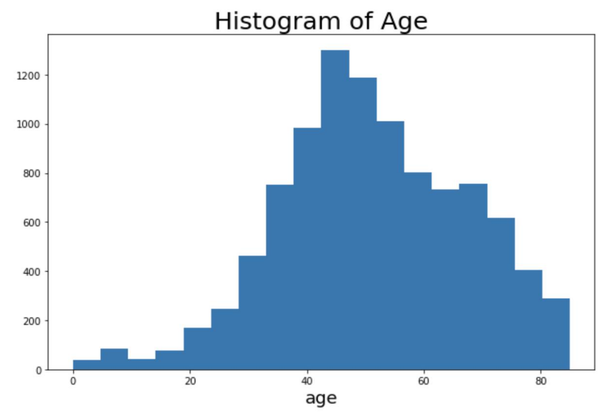
In order to achieve high accuracy and efficiency in the live web app, we choose to apply transfer learning, which is to combine layers in MobileNet except for the output layer with three added layers subsequently.

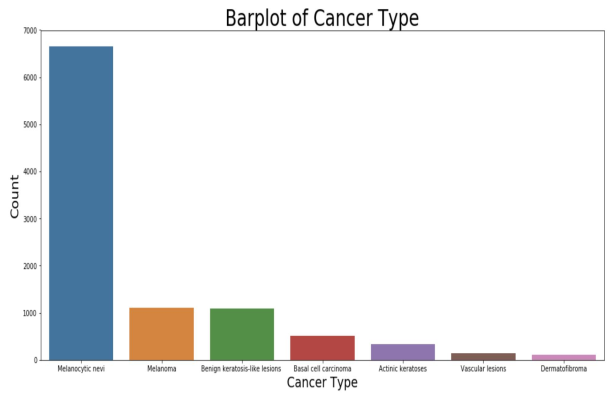
What’s more, we specify several parameters in the training process based on our experiences. The optimization function we use is adam, and loss function is cross entropy. Also, we use accuracy as a metric to evaluate predicting performance, and we set 26 epochs, the batch size is 32, so there are 188 batches in each epoch.

5. Results

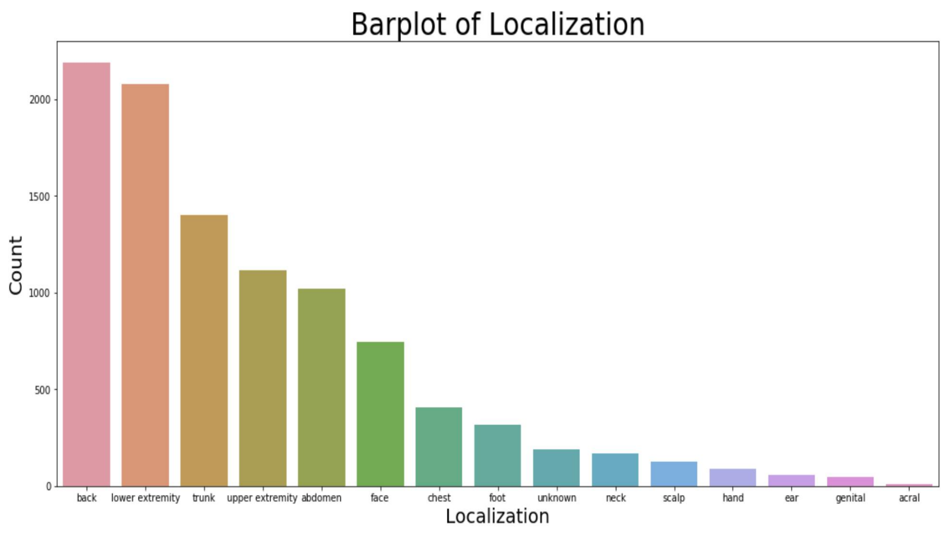
5.1 Exploratory data analysis

As it’s shown in the histogram of age, most of the patients’ age varies from 35 to 70. The bar plot is to visualize the data for each categorical variable. For the cancer type, ‘nv’ type occupies about 66% of the data, indicating that the data set is imbalanced and our model should be able to achieve an accuracy rate of 66% or higher.

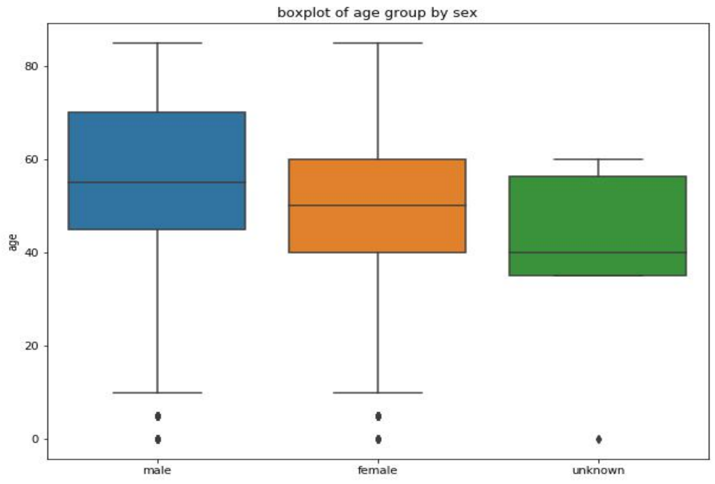


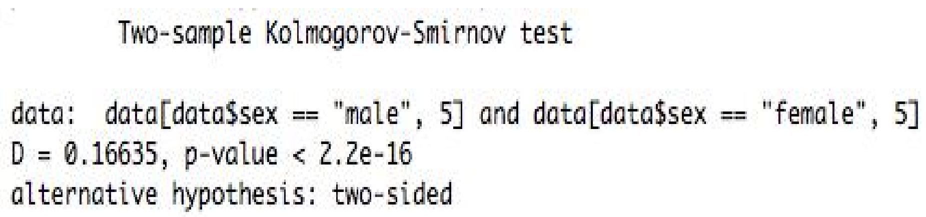


The bar plot of “localization” gives information about common regions of skin cancer. It seems that back, lower extremity, and trunk are the most common regions.

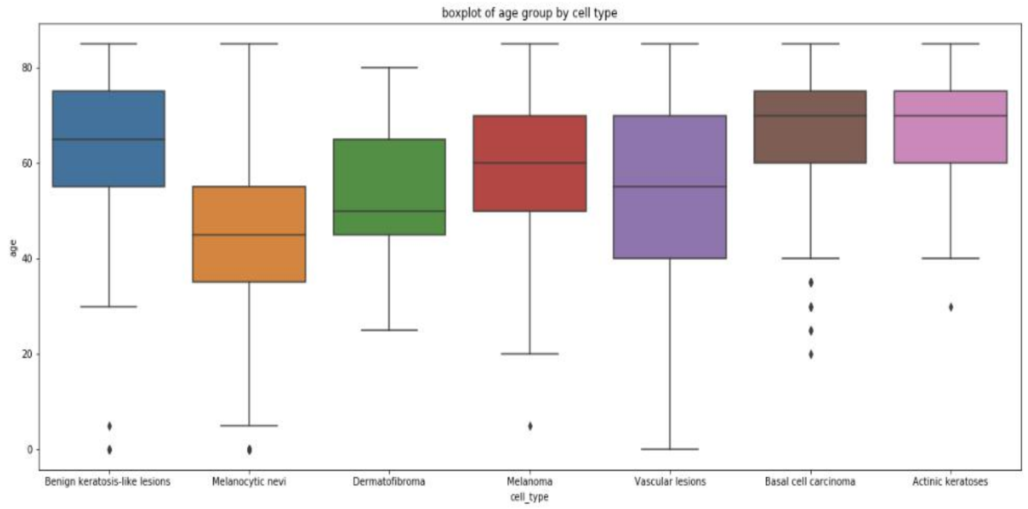


Next, a box plot is drawn to reflect the relationship between sex and age at diagnosis. From the box plot, it is clear that ages at diagnosis are different among sex groups. To confirm this conclusion, we perform Komogorov-Smirnov test with null hypothesis that the male age distribution is the same as female age distribution. Since the p-value for KS test is under 0.05, we conclude that sex has an impact on age distribution.



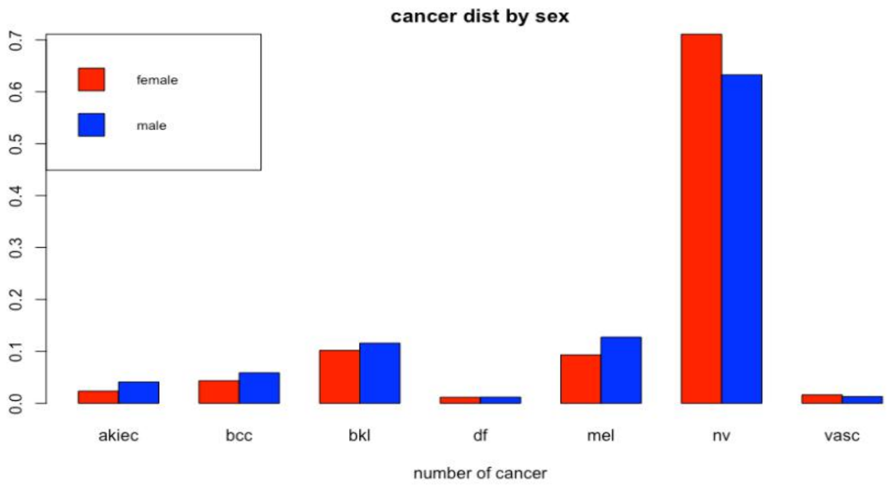

```
        Two-sample Kolmogorov-Smirnov test

data:  data[data$sex == "male", 5] and data[data$sex == "female", 5]
D = 0.16635, p-value < 2.2e-16
alternative hypothesis: two-sided
```

As for the relationship between age at diagnosis and skin cancer type, the chi-square test is conducted and also a box plot is drawn. First, we divide ages into three subgroups, specifying under 30 years old as young, 30 to 60 as middle-aged, and over 60 as old. In the box plot, it is shown that the percentage of age range is obviously different among cells. And the result of chi-square test also confirms our assumption that age and cancer types are dependent variables.



```
            Pearson's Chi-squared test

data:  ta
X-squared = 2106.7, df = 12, p-value < 2.2e-16
```

Besides, we explore the relations between cancer types and sex. The bar plot demonstrates cancer distribution by sex. From the bar plot, the percentage of female is greater than male in cell Melanocytic nevi and Vascular lesions, while lower than or equal to male in other cell types. Combined with the result of chi-square test, we conclude that cell type is dependent of sex.

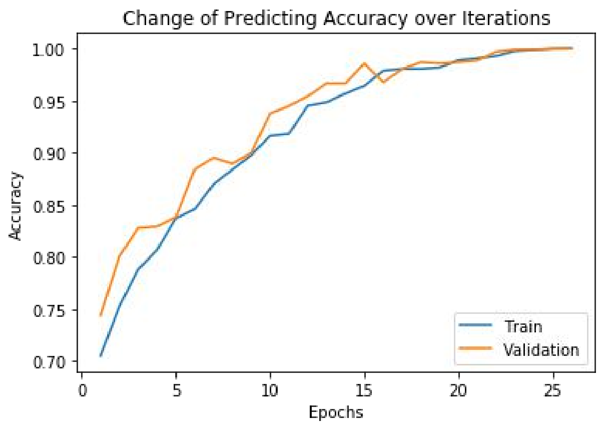


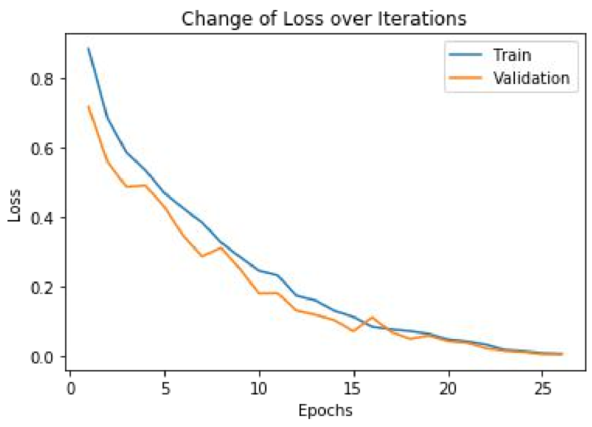
```
      female male
akiec    106  221
bcc      197  317
bkl      463  626
df        52   63
mel      424  689
nv      3237 3421
vasc      73   69

      Pearson's Chi-squared test

data:  ta
X-squared = 89.622, df = 6, p-value < 2.2e-16
```

5.2 CNN model results

After 26 iterations, the model ended with the validation loss being about 0.0044 and accuracy at 100%，Meanwhile, training loss is 0.0062 and accuracy is also 100%. Thus, we decided to stop our training and then apply it to our test data, which results in the test accuracy of 100% and the loss of 0.004.

epochs	train accuracy	train loss	validation accuracy	validation loss
1	0.71	0.88	0.74	0.72
5	0.84	0.47	0.84	0.43
10	0.92	0.25	0.94	0.18
15	0.96	0.11	0.99	0.07
20	0.99	0.05	0.99	0.04
26	1.00	0.01	1.00	0.00





5.3 Web live app

We design a simple app, which requires input of the path of an image to be detected, and will output the corresponding skin cancer type. A video of the app is uploaded at https://drive.google.com/open?id=1FX6T-ZV5QDGD6RYBg3CzK_LTE3yAkRKb.

6. Discussion and conclusion

As it's shown in the previous results, our model has an incredibly high test accuracy of 100%, and we think there are several reasons for this:
1. The whole data set is unbalanced with proportion of one class being close to 66%.
2. Test set only has 1000 images, so perhaps it's not representative
3. In our loss function, it assigns equal weights to 7 classes, which results in undifferentiated treatment.

Therefore, we come up with the following remedies for further improvement:
1. We can use data augmentation to reduce the class imbalance and in so doing get categorical accuracy scores that were not heavily skewed by a single majority class.

2.We can expand our data set by collecting more image data from real life or corresponding website.
3. We may want to assign different weights according to real word situation, indicating that it would penalize different misclassification error to different degrees accordingly.

7. Bibliography

[1] The HAM10000 dataset, Philipp Tschandl, Cliff Rosendahl and Harald Kittler, 2018 Aug 14, retrieved from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6091241/

[2] MobileNet version 2, Matthijs Hollemans, 2018 Apr 22, retrieved from: https://machinethink.net/blog/mobilenet-v2/

8. Appendix

8.1 Codes

8.1.1 Create training, validation, testing set

```
from shutil import copyfile
import pandas as pd
import os
from sklearn.model_selection import train_test_split
data_root = "C:\\Users\\30686\\Desktop\\ADA\\"
data = pd.read_csv(data_root+"HAM10000_metadata.csv")
train,test = train_test_split(data,random_state=1,test_size=1015)
train,validation =
train_test_split(train,random_state=2,train_size=6000)
for i in range(len(train)):
    img = train.iloc[i,1]
    label = train.iloc[i,2]
    if int(img[-5:])>=29306:
        file_path = data_root + "HAM10000_images_part_2\\" + img +
".jpg"
    else:
        file_path = data_root + "HAM10000_images_part_1\\" + img +
".jpg"
    new_path = data_root + "train\\" + label + "\\" + str(i) + ".jpg"
    copyfile(file_path,new_path)
for i in range(len(validation)):
    img = validation.iloc[i,1]
    label = validation.iloc[i,2]
    if int(img[-5:])>=29306:
        file_path = data_root + "HAM10000_images_part_2\\" + img +
".jpg"
    else:
        file_path = data_root + "HAM10000_images_part_1\\" + img +
".jpg"
    new_path = data_root + "validation\\" + label + "\\" + str(i) +
".jpg"
    copyfile(file_path,new_path)
for i in range(len(test)):
    img = test.iloc[i,1]
    label = test.iloc[i,2]
    if int(img[-5:])>=29306:
        file_path = data_root + "HAM10000_images_part_2\\" + img +
".jpg"
    else:
        file_path = data_root + "HAM10000_images_part_1\\" + img +
".jpg"
    new_path = data_root + "test\\" + label + "\\" + str(i) + ".jpg"
    copyfile(file_path,new_path)
```

8.1.2 Construct and train CNN model

```
%tensorflow_version 2.x
import tensorflow as tf
import numpy as np
from pickle import dump
from google.colab import drive
drive.mount('/content/drive')
train_path = '/content/drive/My Drive/ADA CNN/train'
validation_path = '/content/drive/My Drive/ADA CNN/validation'
test_path = '/content/drive/My Drive/ADA CNN/test'
MobileNet =
tf.keras.applications.mobilenet_v2.MobileNetV2(input_shape=(224,224,3),
include_top=False, weights='imagenet',layers=tf.keras.layers)
MobileNet.trainable = False
model_transfer = tf.keras.Sequential([
  MobileNet,
  tf.keras.layers.GlobalAveragePooling2D(),
  tf.keras.layers.Dense(512,activation='relu'),
  tf.keras.layers.Dense(7,activation='softmax')
  ])
```

```
model_transfer.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])
checkpoint_path = "/content/drive/My Drive/ADA CNN/checkpoint.ckpt"
cp_callback =
tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,save_weigh
ts_only=True,verbose=1,save_best_only=True)
reduce_lr = tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss',
factor=0.2, patience=2, min_lr=0.00001)
train_datagen =
tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
train_generator = train_datagen.flow_from_directory(
        train_path,
        target_size=(224, 224),
        batch_size=32)
test_datagen =
tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
validation_generator = test_datagen.flow_from_directory(
        validation_path,
        target_size=(224, 224),
        batch_size=32)
history = model_transfer.fit_generator(
        train_generator,
        steps_per_epoch = 188,
        epochs=30,
        validation_data=validation_generator,
        validation_steps=94,
        callbacks=[cp_callback,reduce_lr],
        verbose=1
        )
with open('/content/drive/My Drive/ADA CNN/train_history', 'wb') as
handle:
    dump(history.history, handle)
test_generator = test_datagen.flow_from_directory(
        test_path,
        target_size=(224, 224),
        batch_size=32)
model_transfer.evaluate_generator(
    test_generator,
    verbose=1
)
```

8.1.3 The plots in CNN model

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import os
import fnmatch
the_path = "/Users/luyue_chen/Documents/ADAproj/ADA/test"
i = -1
count_test = np.zeros(8)
for root, dir, files in os.walk(the_path):
    i = i+1
    k = 0
    for items in fnmatch.filter(files, "*"):
        k = k+1
    print(k)
    count_test[i] = k
count_test = count_test[1:8]
the_path = "/Users/luyue_chen/Documents/ADAproj/ADA/train"
i = -1
count_train = np.zeros(8)
for root, dir, files in os.walk(the_path):
    i = i+1
    k = 0
    for items in fnmatch.filter(files, "*"):
        k = k+1
    print(k)
    count_train[i] = k
count_train = count_train[1:8]
the_path = "/Users/luyue_chen/Documents/ADAproj/ADA/validation"
i = -1
count_valid = np.zeros(8)
for root, dir, files in os.walk(the_path):
    i = i+1
    k = 0
    for items in fnmatch.filter(files, "*"):
        k = k+1
    print(k)
    count_valid[i] = k
count_valid = count_valid[1:8]
prop_test = count_test/np.sum(count_test)
prop_train = count_train/np.sum(count_train)
prop_valid = count_valid/np.sum(count_valid)
n_groups = 7
# create plot
plt.figure()
fig, ax = plt.subplots()
index = np.arange(n_groups)
```

```
bar_width = 0.25
opacity =  0.8

rects1 = plt.bar(index, prop_train, bar_width,
alpha=opacity,
color='lightsteelblue',
label='Train')
rects2 = plt.bar(index + bar_width, prop_valid, bar_width,
alpha=opacity,
color='royalblue',
label='Validation')
rects3 = plt.bar(index+2*bar_width, prop_test, bar_width,
alpha=opacity,
color='darkblue',
label='Test')
plt.xlabel('Skin cancer')
plt.ylabel('proportion')
plt.title('Distribution of data in each class')
plt.xticks(index + bar_width, ('akiec', 'bcc', 'bkl', 'df', 'mel',
'nv', 'vasc'))
plt.legend()
plt.tight_layout()
plt.savefig('Documents/data_distribution.png')
plt.show()
data = pd.read_csv("/Users/luyue_chen/Documents/history.csv")
dat = data.values
plt.plot(dat[:,0],dat[:,1])
plt.plot(dat[:,0],dat[:,3])
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Change of Predicting Accuracy over Iterations')
plt.legend(['Train', 'Validation'], loc='lower right')
plt.plot(dat[:,0],dat[:,2])
plt.plot(dat[:,0],dat[:,4])
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.title('Change of Loss over Iterations')
plt.legend(['Train', 'Validation'], loc='upper right')
```

8.1.4 Web live app

```
import os
os.chdir('D:\\CUSTAT\\5291-Advanced Data Analysis\\proj')
import tensorflow as tf
import matplotlib.pyplot as plt
import numpy as np
from tensorflow import keras
from keras.preprocessing import image
path = input('Please input image path: \n')
def format_example(image):
    #print("Format example called!")
    image = np.expand_dims(img, axis=0)
    # First, convert the data type to tf.float32
    image = tf.cast(image, tf.float32)
    # Second, normalize the image
    image = image / 255.0
    # Third, resize the image
    image = tf.image.resize(image, (224,224))
    return image
im_raw = tf.io.read_file(path)
img = tf.image.decode_jpeg(im_raw, channels=3)
img = image.load_img(path, grayscale=False)
img = np.array(img)
img = format_example(img)
model = keras.models.load_model('new_model.h5')
pred = model.predict(img).tolist()
pred_idx = pred.index(max(pred))+1
skin_dict = {1:'akiec', 2:'bcc', 3:'bkl', 4:'df', 5:'mel', 6:'nv',
7:'vasc'}
print('The skin cancer type is: {}'.format(skin_dict[pred_idx]))
```

8.1.5 Exploratory data analysis

```
### Python code for dataset ###
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('HAM10000_metadata.csv')
df.isnull().sum()
lesion_type_dict = {
    'nv': 'Melanocytic nevi',
    'mel': 'Melanoma',
    'bkl': 'Benign keratosis-like lesions ',
    'bcc': 'Basal cell carcinoma',
    'akiec': 'Actinic keratoses',
    'vasc': 'Vascular lesions',
    'df': 'Dermatofibroma'
}
dx_type_dict = {'histo': 'histopathology',
                  'follow_up': 'follow-up_examination',
                  'consensus':'expert consensus',
                  'confocal': 'in-vivo confocal microscopy'
}
df['cell_type'] = df['dx'].map(lesion_type_dict.get)
df['confirm_type'] = df['dx_type'].map(dx_type_dict.get)
df.head()
print(df.dtypes)
df.isnull().sum()   # 57 entries missing value in age
df['age'].fillna((df['age'].mean()), inplace=True)   #replace by mean
value

#bar plot by cell type
cell = df['cell_type'].value_counts().sort_index()
cell.plot(kind = 'bar')

#bar plot by confirmation
confirm = df['confirm_type'].value_counts().sort_index()
confirm.plot(kind = 'bar')
plt.figure(figsize=(10,8))
sns.boxplot(
    data=df,
    x='sex',
    y='age',
).set_title('boxplot of age group by sex')
plt.figure(figsize=(20,8))
sns.boxplot(
    data=df,
    x='cell_type',
    y='age',
).set_title('boxplot of age group by cell type')
plt.figure(figsize=(10,8))
sns.boxplot(
    data=df,
    x='confirm_type',
    y='age',
).set_title('boxplot of age group by confirm type')
plt.figure(figsize=(10, 6.5))
plt.hist(df["age"], bins = 18)
plt.xlabel("age", fontsize=18)
print(df['cell_type'].value_counts())
plt.figure(figsize=(18,8))
sns.countplot(data = new_dat, x = "cell_type",
order=df['cell_type'].value_counts().index)
print(df['confirm_type'].value_counts())
```

```
plt.figure(figsize=(13,8))
sns.countplot(data = new_dat, x = "confirm_type",
order=df['confirm_type'].value_counts().index)
print(df['localization'].value_counts())
plt.figure(figsize=(18,8))
sns.countplot(data = new_dat, x = "localization",
order=df['localization'].value_counts().index)
print(df['sex'].value_counts())
plt.figure(figsize=(10,8))
sns.countplot(data = new_dat, x = "sex",
order=df['sex'].value_counts().index)

### R code for test only###
data <- read.csv("/Users/yanzhuchen/Desktop/HAM10000_metadata.csv")
ks.test(data[data$sex=="male",5], data[data$sex=="female",5])
sub <- data[!is.na(data$age),]
dim(sub)
sub$range[sub$age<=30] <- 'young'
sub$range[(sub$age>30)&(sub$age<=60)] <- 'middle-aged'
sub$range[sub$age>60] <- 'old'
ta <- table(sub$range, sub$dx)
ta
chisq.test(ta)
barplot(ta, main='age dist by cancer type', xlab='cancer type', col =
c('red', 'blue', 'green'), legend=TRUE, args.legend = list(x="topleft",
cex=0.8))
sub <- subset(data, sex!="unknown")
ta <- table(sub$dx, sub$sex)
ta <- ta[,-3]
ta
ta <- t(ta)
ta[1,] <- ta[1,]/sum(ta[1,])
ta[2,] <- ta[2,]/sum(ta[2,])
chisq.test(ta)
barplot(ta, main = 'cancer dist by sex', xlab = 'number of cancer',
col=c('red','blue'),legend=TRUE, beside=TRUE, args.legend =
list(x="topleft", cex=0.8))
```