

COMP3314 Assignment 3

Image Classification Report

Team name: 99%

Chen Yuxuan

Li Yufei

Liu Yantong 3036127012

April 19, 2025

Contents

1 Abstract

2 Dataset Analysis

In this section, we present an overview of the dataset, analyzing the statistics on the number of categories. Additionally, we will visualize representative examples from each category to illustrate the diversity and characteristics inherent within the dataset. This analysis aims to provide a foundational understanding of the dataset's structure, which is essential for subsequent modeling and interpretation.

2.1 Load Data

We load the data and subsequently convert both the training and testing datasets into NumPy arrays, which are easier for our model to process.

```
csv_path, csv_test_path = "./data/train.csv", "./data/test.csv"
img_dir, img_dir_test = "./data/train_ims", "./data/test_ims"
data_train = pd.read_csv(csv_path)
data_test = pd.read_csv(csv_test_path)
X_train, y_train, X_test, y_test = [], [], [], []

for _, row in data_train.iterrows():
    img_path = os.path.join(img_dir, row.iloc[0])
    label = int(row.iloc[1])
    img = Image.open(img_path).convert("RGB")
    img = np.array(img).flatten()
    X_train.append(img)
    y_train.append(label)

for _, row in data_test.iterrows():
    img_path = os.path.join(img_dir_test, row.iloc[0])
    label = int(row.iloc[1])
    img = Image.open(img_path).convert("RGB")
    img = np.array(img).flatten()
    X_test.append(img)
    y_test.append(label)

X_train, y_train = np.array(X_train), np.array(y_train)
X_test, y_test = np.array(X_test), np.array(y_test)
X_train_flatten = X_train.reshape(X_train.shape[0], -1)
X_test_flatten = X_test.reshape(X_test.shape[0], -1)
```

Listing 1: Load Data

2.2 Statistics on the number of categories

2.2.1 Dataset Size

```
print("The shape of the training set: ", X_train.shape)
print("The shape of the testing set: ", X_test.shape)
print("Training set (flattened): ", X_train_flatten.shape)
print("Testing set (flattened): ", X_test_flatten.shape)
print("The size of the label of the training set: ", y_train.shape)
print("The size of the label of the testing set: ", y_test.shape)
```

Listing 2: Print the shape of the datasets

- **Size Statistics**
- The shape of the training set: (50000, 32, 32, 3)
- The shape of the testing set: (10000, 32, 32, 3)
- Training set (flattened): (50000, 3072)
- Testing set (flattened): (10000, 3072)
- The size of the label of the training set: (50000,)
- The size of the label of the test set: (10000,)

2.2.2 Category Label Count

```
# Count the labels
count_labels = np.unique(y_train, return_counts=True)

# Create DataFrame
label_counts_df = pd.DataFrame({
    "Label": count_labels[0],
    "Count": count_labels[1]
}).set_index("Label")

# Print DataFrame
print("Count of each label:")
print(label_counts_df)
```

Listing 3: Count the number of each category

Label 0	Label 1	Label 2	Label 3	Label 4
5038	5016	5032	4991	4982
Label 5	Label 6	Label 7	Label 8	Label 9
4967	4985	4998	5002	4989

Table 1: Label Counts in training set

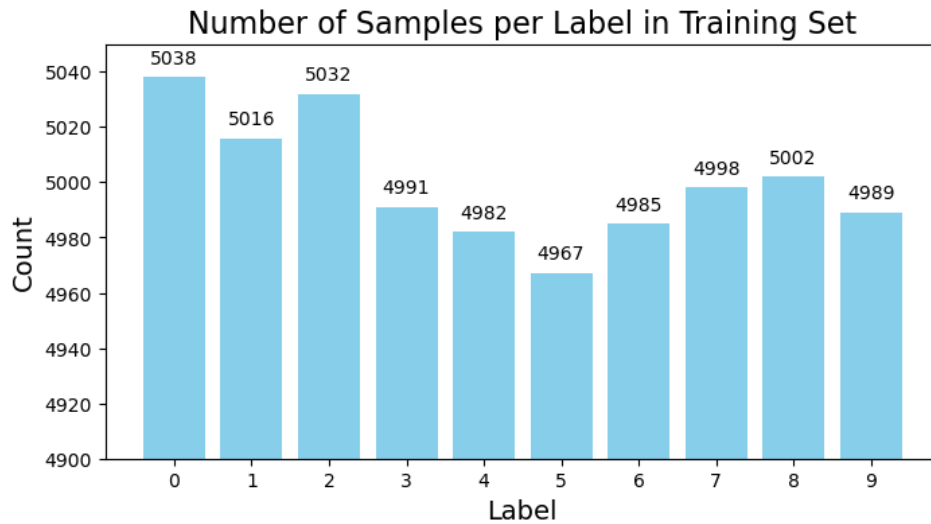


Figure 1: Label Counts in training set

2.2.3 Other Statistics

```
print("Other Statistics for the training set label:")
print(pd.Series(y_train).describe())
```

Listing 4: Print other statistics of the labels of the training set

```
count    50000.00000
mean         4.49258
std         2.87539
min         0.00000
25%         2.00000
50%         4.00000
75%         7.00000
max         9.00000
```

2.3 Image Visualization

2.3.1 One Example for Each Category

```
def visualize_images_by_label(img_dir, data, y):
    unique_labels = np.unique(y)
    fig, axes = plt.subplots(2, 5, figsize=(15, 6))
    for i, label in enumerate(unique_labels[:10]):
        indices = np.where(y == label)[0]
        idx = np.random.choice(indices)
        img = Image.open(os.path.join(img_dir, data.iloc[idx, 0]))
        row = i // 5
        col = i % 5
        ax = axes[row, col]
        ax.imshow(img)
        ax.set_title(f"Label {label}")
        ax.axis('off')
    for j in range(i+1, 10):
        row = j // 5
        col = j % 5
        axes[row, col].axis('off')
    plt.tight_layout()
    plt.savefig("1 example.png", format='png', bbox_inches='tight')
    plt.show()
visualize_images_by_label(img_dir, data_train, y_train)
```



Figure 2: Visualization of one example for each category

2.3.2 Ten Examples for Each Category

```

count_labels = np.unique(y_train, return_counts=True)
label_counts_df = pd.DataFrame({
    "Label": count_labels[0],
    "Count": count_labels[1]
}).set_index("Label")

def get_example_images(label, num_examples=10):
    indices = np.where(y_train == label)[0]
    selected_indices = np.random.choice(indices,
        size=min(num_examples, len(indices)), replace=False)
    example_images = [os.path.join(img_dir, data_train.iloc[i, 0])
        for i in selected_indices]
    return " ".join([f"<img src='{img}' width='50' />"
        for img in example_images])

label_counts_df["Example Images"] =
label_counts_df.index.map(get_example_images)
HTML(label_counts_df.to_html(escape=False))

```





































































































	Count	Example Images									
Label											
0	5038										
1	5016										
2	5032										
3	4991										
4	4982										
5	4967										
6	4985										
7	4998										
8	5002										
9	4989										

Figure 3: Visualization of ten example for each category

3 Image Preprocessing

TBD

3.1 Image Resizing

TBD

3.2 Feature Extraction

TBD

4 Model Training

TBD

4.1 Model 1

TBD

4.2 Model 2

TBD

4.3 Model 3

TBD

```
def add(x,y):  
    res = x + y  
    printf("{x} + {y} = {res}")  
    return res
```

```
x = 3  
y = 2  
r = add(x,y)  
print(r)
```

Listing 5: Python sample code



Figure 4: woshinailong

5 Results

TBD