Bottle Segregation

Biologically Inspired Artificial Intelligence

Krzysztof Czarnecki

June 6, 2023

Silesian University of Technology

Table of Contents

Project goals

- The aim of the project is to train a model capable of sorting bottles

Project goals

- The aim of the project is to train a model capable of sorting bottles
- The model receives a picture of the bottle, after that it determines what type the bottle is

Project goals

- The aim of the project is to train a model capable of sorting bottles
- The model receives a picture of the bottle, after that it determines what type the bottle is
- There are be 5 types of bottles: beer, plastic, soda, water and wine

Silesian University of Technology

Used dataset

1 Introduction

- The ”Bottles Synthetic Images” dataset from kaggle.com was used to train the model
- The dataset contains 5,000 photos for each type of bottle, which gives a total of 25,000

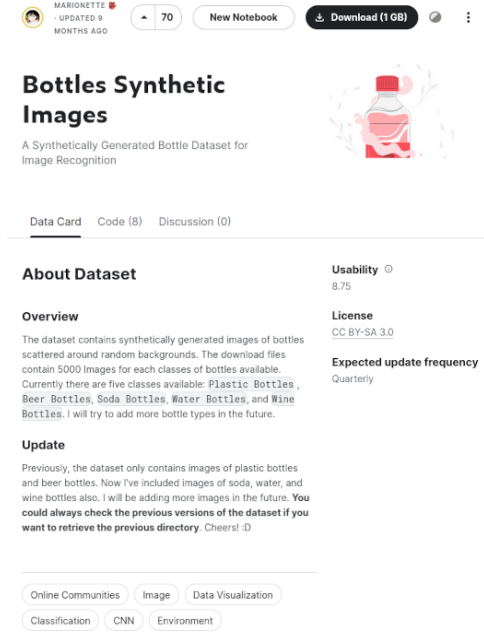


Table of Contents

2 Model training

Architecture

The model will be trained using the ResNet18 architecture, a popular convolutional neural network (CNN) model.
Some information:

- ResNet18 is a deep CNN architecture known for its effectiveness in image classification tasks
- It consists of 18 layers, including convolutional layers, pooling layers, and fully connected layers
- The residual connections in ResNet18 help in alleviating the vanishing gradient problem during training.

Silesian University of Technology

Data Preprocessing

2 Model training

- Before training the model, the collected dataset underwent preprocessing steps to ensure its quality and compatibility with the model
- Preprocessing steps included resizing the images to a consistent resolution, normalizing the pixel values, and splitting the dataset into training and validation sets.

In [7]:
```
def get_mean_and_std(loader):
    mean = 0.
    std = 0.
    total_images_count = 0
    for images, _ in loader:
        image_count_in_a_batch = images.size(0)
        images = images.view(image_count_in_a_batch, images.size(1), -1)
        mean += images.mean(2).sum(0)
        std += images.std(2).sum(0)
        total_images_count += image_count_in_a_batch

    mean /= total_images_count
    std /= total_images_count

    return mean, std
```

In [8]:
```
mean_and_std = get_mean_and_std(training_loader)
print(mean_and_std)
```

```
(tensor([0.4729, 0.4099, 0.3521]), tensor([0.1786, 0.1670, 0.1610]))
```

In [4]:
```
training_transforms = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
    transforms.ToTensor(),
    transforms.Normalize(mean_and_std[0], mean_and_std[1])
])

validation_transforms = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean_and_std[0], mean_and_std[1])
])
```

Data Preprocessing

2 Model training

Splitting into Training and Validation Sets: Preprocessing also involves splitting the dataset into training and validation sets. The training set is used to train the model, while the validation set is used to evaluate its performance. This separation helps us assess how well the model generalizes to new, unseen data.

In [2]:
```
input_folder ='./Bottle Images'
splitfolders.ratio(input_folder, output = "splitted_bootles", seed = 42, ratio = (.7, .2, .1), group_prefix = None)
```

Photos for training: 70%, for validation: 20%, for test by user: 10%.

Silesian University of Technology

Optimizer

2 Model training

- optim.SGD - optimizer in PyTorch
- SGD is a popular and effective optimizer in the field of machine learning

Key variables in optim.SGD:

- Learning Rate
- Momentum
- Weight Decay

```
optimizer = optim.SGD(resnet18_model.parameters(), lr = 0.01, momentum = 0.9, weight_decay = 0.003)
```

Loss function

CrossEntropyLoss() is designed to calculate the cross-entropy loss between the predicted class probabilities and the true class labels. It is particularly suitable for multi-class classification problems. The function takes two inputs:

1. Predicted Class Probabilities: The output of the ResNet18 model is a vector of predicted class probabilities for each input image. These probabilities represent the model's confidence scores for each class.
2. True Class Labels: The true class labels represent the ground truth information for each input image. These labels indicate the correct class of the image.

Silesian University of Technology

Training function

2 Model training

```python
def train_nn(model, best_acc, training_loader, validation_loader, criterion, optimizer, n_epochs):
    device = set_device()

    for epoch in range(n_epochs):
        print("Epoch number %d" % (epoch + 1))
        model.train()
        running_loss = 0.
        running_correct = 0.
        total = 0

        for data in training_loader:

            images, labels = data

            images = images.to(device)

            labels = labels.to(device)

            total += labels.size(0)

            optimizer.zero_grad()

            outputs = model(images)

            _, predicted = torch.max(outputs.data, 1)

            loss = criterion(outputs, labels)

            loss.backward()

            optimizer.step()

            running_loss += loss.item()
            running_correct += (labels == predicted).sum().item()


        epoch_loss = running_loss/len(training_loader)
        epoch_acc = 100.0 * running_correct / total

        print("    - Training dataset. Got %d out of %d images correctly (%.3f%%). Epoch loss: %.3f"
              % (running_correct, total, epoch_acc, epoch_loss))

        test_dataset_acc = evaluate_model_on_validation_set(model, validation_loader)

        if(test_dataset_acc > best_acc):
            best_acc = test_dataset_acc
            save_best_state(model, epoch, optimizer, best_acc)

    print("    Finished")
```

```python
def evaluate_model_on_validation_set(model, validation_loader):
    model.eval()
    predicted_correctly_on_epoch = 0
    total = 0
    device = set_device()

    with torch.no_grad():
        for data in validation_loader:
            images, labels = data
            images = images.to(device)
            labels = labels.to(device)
            total += labels.size(0)
            outputs = model(images)
            _, predicted = torch.max(outputs.data, 1)
            predicted_correctly_on_epoch += (labels == predicted).sum().item()
    epoch_acc = 100.0 * predicted_correctly_on_epoch / total
    print("    - Testing dataset. Got %d out of %d images correctly (%.3f%%)"
          % (predicted_correctly_on_epoch, total, epoch_acc))
    return epoch_acc
```

```python
def save_best_state(model, epoch, optimizer, best_acc):
    state = {
        'epoch': epoch + 1,
        'model': model.state_dict(),
        'best_accuracy': best_acc,
        'optimizer': optimizer.state_dict(),
        'date': datetime.now()
    }
    torch.save(state, 'best_model.pth.tar')
```

```
91.64
Epoch number 1
    - Training dataset. Got 16440 out of 17500 images correctly (93.943%). Epoch loss: 0.183
    - Testing dataset. Got 4680 out of 5000 images correctly (93.600%)
Epoch number 2
    - Training dataset. Got 16537 out of 17500 images correctly (94.497%). Epoch loss: 0.164
    - Testing dataset. Got 4330 out of 5000 images correctly (86.600%)
    Finished
1
93.6
```

Table of Contents

Purpose of this app

The application's task is to read the image from the camera, send the photo to the model and display the effect to the user.

Silesian University of Technology

How to achieve it?

3 Application

+

Silesian University of Technology

Result

3 Application

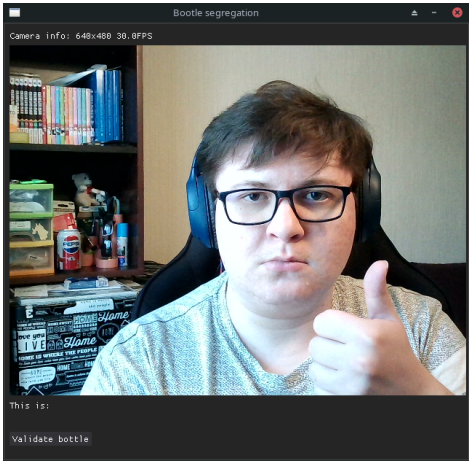


Table of Contents

Silesian University of Technology

Photos from the internet

4 Tests


```
In [9]: path = 'beer.jpg'
In [10]: classify(model, image_transforms, path, classes)
```

Beer Bottle

```
In [11]: path = 'wine.jpg'
In [12]: classify(model, image_transforms, path, classes)
```

Wine Bottle

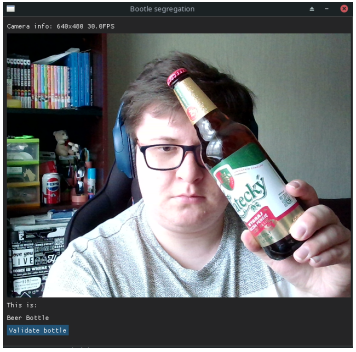
```
In [13]: path = 'plastic.jpg'
In [14]: classify(model, image_transforms, path, classes)
```

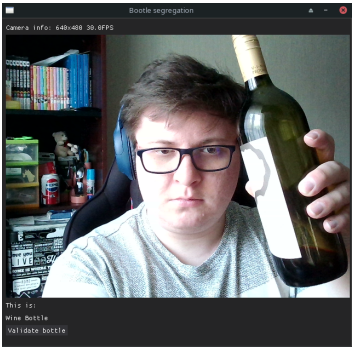
Plastic Bottle

Silesian University of Technology

Photos from the camera

4 Tests





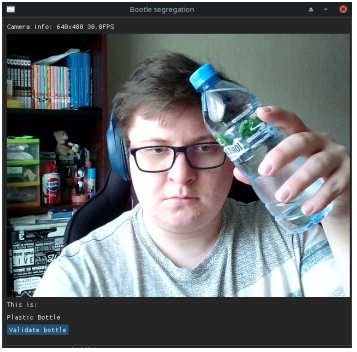


Table of Contents

Silesian University of Technology

What could be improved?

5 Conclusions

- addition of 6 type - no bottle
- better data for soda and beer bottles

Bottle Segregation

Thank you for listening!

Any questions?