Nhóm 11

Trần Thanh Khoa

Thái Đức Toàn

Mai Đức Khiêm

I: Mô tả phương pháp Template Matching

Template Matching là một kỹ thuật trong xử lý ảnh dùng để tìm và so khớp một mẫu (template) trong một ảnh lớn hơn (target image). Phương pháp này thường được sử dụng để phát hiện các đối tượng trong ảnh hoặc phân loại các mẫu dữ liệu. Cụ thể hơn, trong bài toán phân loại rượu vang, chúng ta áp dụng phương pháp Template Matching dựa trên khoảng cách Euclid giữa các mẫu đặc trưng.

Các bước cơ bản trong Template Matching:

1. **Chuẩn bị ảnh mẫu và ảnh mục tiêu**:
    - **Template (ảnh mẫu)**: Đây là một phần nhỏ của ảnh hoặc dữ liệu mà bạn muốn tìm kiếm trong một ảnh lớn hơn. Ví dụ: một phần nhãn chai rượu hoặc các đặc tính hóa học của một loại rượu cụ thể.
    - **Target Image (ảnh mục tiêu)**: Đây là ảnh lớn hơn chứa đối tượng mà bạn muốn tìm kiếm. Trong bài toán này, ảnh mục tiêu là tập dữ liệu chứa đặc trưng của nhiều loại rượu khác nhau.
2. **Di chuyển và so khớp template trong ảnh mục tiêu**:
    - **Template Matching** hoạt động bằng cách di chuyển ảnh mẫu qua toàn bộ ảnh mục tiêu (hoặc dữ liệu) để tìm vùng có sự tương đồng cao nhất.
    - Tại mỗi vị trí, thuật toán sẽ tính toán độ tương đồng giữa template và vùng tương ứng trong ảnh mục tiêu. Có nhiều phương pháp tính độ tương đồng, trong đó phổ biến là:
        - **Sum of Squared Differences (SSD)**: Tính tổng bình phương sai số giữa các điểm ảnh.
        - **Normalized Cross-Correlation (NCC)**: Tính toán mức độ tương quan giữa template và vùng tương ứng.
3. **Xác định vị trí tối ưu**:
    - Vị trí trong ảnh mục tiêu mà độ tương đồng giữa template và vùng ảnh là cao nhất (tức là tương quan lớn nhất hoặc sự khác biệt nhỏ nhất) sẽ được coi là nơi template xuất hiện trong ảnh mục tiêu.
    - Phương pháp **Euclidean distance** (khoảng cách Euclid) cũng được sử dụng để đo độ gần gũi giữa các mẫu trong không gian đặc trưng. Trong bài toán này, khoảng cách Euclid được tính giữa các vector đặc trưng của rượu.
4. **Kết quả**:
    - Kết quả trả về là tọa độ hoặc lớp (class) của mẫu trong ảnh hoặc dữ liệu mục tiêu có sự tương đồng cao nhất với template.

- Trong bài toán phân loại rượu vang, phương pháp này giúp xác định loại rượu nào gần nhất với mẫu huấn luyện dựa trên khoảng cách Euclid, từ đó phân loại mẫu kiểm tra vào lớp tương ứng.

II: Mô tả cơ sở dữ liệu:

Cơ sở dữ liệu được sử dụng trong bài toán này là bộ dữ liệu **"Wine-Classifier-Italy"**, bao gồm các thông tin về đặc tính hóa học của các mẫu rượu vang Ý. Đây là bộ dữ liệu phổ biến trong các bài toán học máy liên quan đến phân loại nhiều lớp (multiclass classification).

Thông tin chi tiết về bộ dữ liệu:

- **Số lượng mẫu (Instances)**: 178 mẫu rượu.
- **Số lượng lớp (Labels)**: Có 3 lớp, tương ứng với 3 loại rượu vang khác nhau từ ba giống nho (cultivars) khác nhau.
- **Số lượng đặc trưng (Features)**: 13 đặc trưng hóa học của mỗi mẫu rượu, bao gồm:
    1. **Alcohol** – Nồng độ cồn.
    2. **Malic acid** – Hàm lượng acid malic.
    3. **Ash** – Hàm lượng tro.
    4. **Alcalinity of ash** – Độ kiềm của tro.
    5. **Magnesium** – Hàm lượng magiê.
    6. **Total phenols** – Tổng lượng phenol.
    7. **Flavanoids** – Hàm lượng flavonoid.
    8. **Nonflavanoid phenols** – Hàm lượng phenol không phải flavonoid.
    9. **Proanthocyanins** – Hàm lượng proanthocyanin.
    10. **Color intensity** – Cường độ màu.
    11. **Hue** – Tông màu.
    12. **OD280/OD315 of diluted wines** – Tỷ lệ hấp thụ ánh sáng ở 280 nm và 315 nm.
    13. **Proline** – Hàm lượng proline.

Phân bố dữ liệu theo các lớp:

Mỗi mẫu rượu được gán nhãn từ 1 đến 3, tương ứng với một trong ba loại rượu vang. Sự phân bố này có thể không đồng đều, nhưng mỗi lớp sẽ có một tập hợp các mẫu đại diện cho từng giống nho khác nhau.

Cách chọn tập train và tập test:

1. Phương pháp phân chia ngẫu nhiên (Random Split):

- **Tập huấn luyện (Training Set)**: Sử dụng để huấn luyện mô hình. Tập huấn luyện thường chiếm khoảng **70-80%** tổng số mẫu.
- **Tập kiểm thử (Test Set)**: Sử dụng để đánh giá mô hình, chiếm khoảng **20-30%** tổng số mẫu.

Trong bài toán này, có thể chọn một mẫu ngẫu nhiên từ mỗi lớp để làm tập kiểm tra, còn lại sẽ sử dụng cho tập huấn luyện. Điều này đảm bảo rằng mỗi lớp đều có ít nhất một mẫu để kiểm tra.

2. Phân tầng dữ liệu (Stratified Sampling):

Để đảm bảo rằng tỷ lệ phân bố các lớp trong cả tập huấn luyện và tập kiểm thử được giữ cân bằng, ta có thể sử dụng phương pháp phân tầng. Với phương pháp này:

- **Mỗi lớp** sẽ có một tỷ lệ tương ứng trong cả hai tập huấn luyện và tập kiểm tra. Điều này giúp mô hình không bị thiên lệch về một lớp nhất định, đảm bảo rằng mô hình có cơ hội học đều từ các lớp khác nhau.

Ví dụ:

- Nếu có 178 mẫu và 3 lớp, một cách tiếp cận có thể là chọn 80% dữ liệu của mỗi lớp làm tập huấn luyện và 20% còn lại làm tập kiểm tra. Như vậy, nếu lớp 1 có 59 mẫu, ta chọn 47 mẫu cho tập huấn luyện và 12 mẫu cho tập kiểm tra, tương tự cho các lớp còn lại.

III: Tiến hành phân loại rượu vang Ý

Trong bài toán này, chúng ta sử dụng phương pháp **Template Matching** kết hợp với **Khoảng cách Euclid** để phân loại rượu vang Ý dựa trên các đặc tính hóa học của chúng. Quá trình phân loại sẽ được thực hiện bằng cách tìm kiếm mẫu huấn luyện gần nhất với mẫu kiểm tra và dựa vào đó để dự đoán lớp (class) của mẫu kiểm tra.

Cách làm:

1. **Xác định vấn đề và mục tiêu**:
   - Vấn đề đặt ra là phân loại các loại rượu vang Ý dựa trên các đặc tính hóa học của chúng.
   - Tập dữ liệu bao gồm 178 mẫu rượu từ ba giống nho khác nhau, với 13 đặc tính hóa học được đo lường cho mỗi mẫu.
   - Mục tiêu là xây dựng một mô hình dựa trên phương pháp Template Matching để phân loại rượu vang vào một trong ba loại dựa trên các đặc trưng của nó.
2. **Chuẩn bị dữ liệu**:
   - **Tải dữ liệu**: Tải bộ dữ liệu rượu vang từ thư viện và đọc dữ liệu từ file.
   - **Chia dữ liệu thành tập huấn luyện và tập kiểm tra**: Chọn ngẫu nhiên một mẫu từ mỗi lớp làm tập kiểm tra, còn lại là tập huấn luyện.
   - **Chuẩn hóa dữ liệu**: Các đặc trưng được chuẩn hóa để đảm bảo rằng tất cả các đặc tính có cùng quy mô và không ảnh hưởng đến khoảng cách Euclid.
3. **Xây dựng mô hình Template Matching**:
   - Sử dụng **khoảng cách Euclid** để tính độ tương đồng giữa mẫu kiểm tra và các mẫu huấn luyện.
   - Mẫu huấn luyện nào có khoảng cách nhỏ nhất với mẫu kiểm tra sẽ được chọn làm mẫu gần nhất, và nhãn của nó sẽ được dùng làm dự đoán cho mẫu kiểm tra.
4. **Dự đoán và đánh giá hiệu suất**:
   - Sử dụng mô hình đã huấn luyện để dự đoán nhãn cho tập kiểm tra.
   - Đánh giá mô hình bằng cách tính độ chính xác (accuracy), tức là tỷ lệ phần trăm các dự đoán đúng.

CODE:

```
# Import các thư viện cần thiết
import pandas as pd  # Thư viện để thao tác với dữ liệu dạng bảng
```

```
from sklearn.metrics.pairwise import euclidean_distances  # Hàm tính
khoảng cách Euclid giữa các vector
from sklearn.preprocessing import StandardScaler  # Thư viện để chuẩn hóa
dữ liệu

# Bước 1: Chuẩn bị dữ liệu
# Đọc dữ liệu từ file CSV chứa thông tin về rượu vang
url = "./wine.data"  # Đường dẫn tới file dữ liệu
# Đặt tên cho các cột trong dữ liệu
columns = ['Class', 'Alcohol', 'Malic_Acid', 'Ash', 'Alcalinity_of_Ash',
'Magnesium', 'Total_Phenols', 'Flavanoids', 'Nonflavanoid_Phenols',
           'Proanthocyanins', 'Color_Intensity', 'Hue',
'OD280_OD315_of_Diluted_Wines', 'Proline']
# Đọc dữ liệu vào dataframe
data = pd.read_csv(url, header=None, names=columns)
print(data.head())  # Hiển thị 5 dòng đầu tiên của dữ liệu

# Bước 2: Chia dữ liệu thành tập huấn luyện và kiểm tra
# Định nghĩa hàm để phân chia dữ liệu
def split_data(data):
    # Tạo dataframe rỗng cho tập huấn luyện và kiểm tra
    train_data = pd.DataFrame()
    test_data = pd.DataFrame()

    # Lặp qua từng lớp dữ liệu
    for label in data['Class'].unique():
        class_data = data[data['Class'] == label]  # Lọc dữ liệu theo từng
lớp
        test_sample = class_data.sample(n=1)  # Lấy ngẫu nhiên 1 mẫu từ
mỗi lớp làm tập kiểm tra
        train_samples = class_data.drop(test_sample.index)  # Phần còn lại
làm tập huấn luyện

        # Kết hợp các mẫu huấn luyện và kiểm tra
        test_data = pd.concat([test_data, test_sample])
        train_data = pd.concat([train_data, train_samples])

    return train_data, test_data  # Trả về tập huấn luyện và tập kiểm tra

# Gọi hàm để chia dữ liệu
train_data, test_data = split_data(data)

# Bước 3: Chuẩn hóa dữ liệu
# Chuẩn hóa dữ liệu giúp đưa các đặc trưng về cùng một đơn vị đo để mô
hình học hiệu quả hơn
```

```
scaler = StandardScaler()  # Khởi tạo bộ chuẩn hóa

# Chuẩn hóa các đặc trưng của tập huấn luyện
train_features = scaler.fit_transform(train_data.drop('Class', axis=1))
# Chuẩn hóa tập kiểm tra dựa trên thông số của tập huấn luyện
test_features = scaler.transform(test_data.drop('Class', axis=1))

# Bước 4: Xây dựng mô hình Template Matching
# Định nghĩa hàm phân loại bằng phương pháp template matching
def classify_template_matching(train_features, train_labels,
test_features):
    results = []  # Tạo danh sách lưu kết quả dự đoán

    # Lặp qua từng mẫu kiểm tra
    for idx, test_sample in enumerate(test_features):
        # Tính khoảng cách Euclid giữa mẫu kiểm tra và tất cả các mẫu huấn
luyện
        distances = euclidean_distances([test_sample], train_features)
        # Tìm mẫu huấn luyện gần nhất
        nearest_index = distances.argmin()
        # Dự đoán lớp của mẫu kiểm tra dựa trên mẫu gần nhất
        predicted_class = train_labels.iloc[nearest_index]
        # Thêm dự đoán vào danh sách kết quả
        results.append(predicted_class)

    return results  # Trả về danh sách các kết quả dự đoán

# Tách nhãn lớp từ tập huấn luyện và tập kiểm tra
train_labels = train_data['Class']
test_labels = test_data['Class']

# Gọi hàm để phân loại và lưu kết quả vào predictions
predictions = classify_template_matching(train_features, train_labels,
test_features)

# Bước 5: Đánh giá mô hình
# Tính số lượng dự đoán chính xác
correct = sum(true == pred for true, pred in zip(test_labels,
predictions))
# Tính độ chính xác của mô hình
accuracy = correct / len(test_labels)
print(f"Accuracy: {accuracy:.2f}")  # In ra độ chính xác

# Nhận xét kết quả
for true, pred in zip(test_labels, predictions):
```

```
    print(f"Actual: {true}, Predicted: {pred}")  # In ra kết quả từng mẫu
```

Kết quả:

```
   Class  Alcohol  Malic_Acid   Ash  Alcalinity_of_Ash  Magnesium  \
0      1    14.23        1.71  2.43               15.6        127
1      1    13.20        1.78  2.14               11.2        100
2      1    13.16        2.36  2.67               18.6        101
3      1    14.37        1.95  2.50               16.8        113
4      1    13.24        2.59  2.87               21.0        118

   Total_Phenols  Flavanoids  Nonflavanoid_Phenols  Proanthocyanins  \
0           2.80        3.06                  0.28             2.29
1           2.65        2.76                  0.26             1.28
2           2.80        3.24                  0.30             2.81
3           3.85        3.49                  0.24             2.18
4           2.80        2.69                  0.39             1.82

   Color_Intensity   Hue  OD280_OD315_of_Diluted_Wines  Proline
0             5.64  1.04                          3.92     1065
1             4.38  1.05                          3.40     1050
2             5.68  1.03                          3.17     1185
3             7.80  0.86                          3.45     1480
4             4.32  1.04                          2.93      735
Accuracy: 1.00
Actual: 1, Predicted: 1
Actual: 2, Predicted: 2
Actual: 3, Predicted: 3
```

Nhận xét:

1. **Độ chính xác 100%**:
   - Trong trường hợp này, độ chính xác của mô hình đạt 100%, có nghĩa là mô hình đã phân loại đúng tất cả các mẫu trong tập kiểm tra. Đây là một kết quả lý tưởng và cho thấy rằng mô hình hoạt động tốt trên tập dữ liệu đã được chuẩn bị.
   - Tuy nhiên, cần lưu ý rằng độ chính xác 100% có thể là kết quả của việc tập kiểm tra quá nhỏ hoặc không đủ đa dạng, nên không đảm bảo rằng mô hình sẽ tổng quát tốt trên các tập dữ liệu khác.
2. **Dữ liệu kiểm tra nhỏ**:
   - Kết quả cho thấy mỗi lớp chỉ có một mẫu trong tập kiểm tra. Điều này làm giảm tính đại diện của tập kiểm tra, có thể không phản ánh được đầy đủ khả năng phân loại của mô hình.
   - Để có kết quả đáng tin cậy hơn, nên sử dụng thêm các phương pháp như **k-fold cross-validation** để đánh giá hiệu suất mô hình trên các tập kiểm tra khác nhau.
3. **Phân tầng lớp dữ liệu**:

- Phương pháp phân chia dữ liệu đã đảm bảo rằng mỗi lớp đều có mẫu trong cả tập huấn luyện và kiểm tra. Điều này giúp tránh hiện tượng mô hình chỉ học tốt một lớp mà không tổng quát được cho các lớp khác.

Tổng kết:

- Mô hình **Template Matching** hoạt động hiệu quả trên tập dữ liệu này, đạt độ chính xác cao. Tuy nhiên, do tập kiểm tra nhỏ, cần thận trọng trong việc kết luận về khả năng tổng quát hóa của mô hình.
- Để cải thiện, có thể áp dụng các phương pháp kiểm tra khác nhau, chẳng hạn như **cross-validation**, để có cái nhìn khách quan hơn về hiệu suất của mô hình trên nhiều tập dữ liệu khác nhau.