Universidade do Estado do Rio de Janeiro
Instituto Politécnico do Rio de Janeiro

Gustavo de Souza Curty
202110049111

Visão Computacional: Relatório EP3
Professor(a): Silvia Dias

Nova Friburgo 2024

Sumário

1 Introdução

Este relatório apresenta as etapas desenvolvidas no trabalho EP3, focado na segmentação de objetos de interesse e classificação com base nos resultados segmentados. O trabalho foi dividido em duas partes principais:

- **EP3.1**: Segmentação dos objetos de interesse, incluindo o uso de um método automático e a criação de *ground-truth* manual.
- **EP3.2**: Avaliação do método de classificação utilizando os resultados das segmentações obtidas.

2 EP3.1: Segmentação do Objeto de Interesse

2.1 Objetivo

O objetivo desta etapa ë segmentar os objetos de interesse nas imagens, produzindo imagens binárias (0 para o fundo e 1 para o objeto). Dois métodos foram empregados:

- **Manual**: Geração do *ground-truth* para 15% das amostras utilizando o software ImageJ.
- **Automático**: Implementação de um algoritmo baseado no *thresholding* de Otsu.

2.2 Ferramentas Utilizadas

As ferramentas e bibliotecas utilizadas incluem:

- **Python**: Para implementação do algoritmo automático.
- **Bibliotecas**: `numpy`, `opencv-python`, `scikit-image`, `pillow-heif`.
- **ImageJ**: Para criação do *ground-truth* manual.

2.3 Metodologia

2.3.1 Segmentação Automática

Os passos seguidos no método automático foram:

1. Conversão das imagens para tons de cinza.
2. Aplicação do *thresholding* de Otsu para gerar máscara binária.
3. Salvamento das máscaras geradas em um diretório específico.

2.3.2 Segmentação Manual

A geração do *ground-truth* seguiu os passos:

1. Abrir a imagem no software ImageJ: File - Open.
2. Selecione o modo Polygon Selection para delimitar o objeto de interesse.
3. Preencha o objeto selecionado: Edit - Fill.
4. Remova o fundo: Edit - clear outside.
5. Transforme a imagem final em Binária: Process - Binary - Make Binary.
6. E por fim, salve a imagem: File - Save As : jpeg.

2.4 Metodologia: Thresholding de Otsu

2.4.1 Definição do Método

O *thresholding* de Otsu ë um algoritmo de segmentação baseado em histograma. Ele busca automaticamente um limiar que minimiza a variância intraclasse entre o fundo e os objetos presentes na imagem.

2.4.2 Como Funciona

O funcionamento do método pode ser descrito em três etapas principais:

1. **Cálculo do Histograma**: O histograma da imagem ë utilizado para contar a frequência de cada nível de intensidade (0 a 255 para imagens em tons de cinza).

2. **Busca do Limiar Otimizado**: O algoritmo varre todos os valores possíveis de limiar, calculando para cada um a variância intraclasse como:

$$\sigma^2_{\text{intraclasse}}(t) = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t), \tag{1}$$

onde ω_1 e ω_2 são as proporções de pixels no fundo e no objeto, respectivamente, e σ_1^2 e σ_2^2 são as variâncias correspondentes.

3. **Aplicação do Limiar**: O limiar que minimiza a variância intraclasse ë escolhido e aplicado para transformar a imagem em binária.

2.5 Resultados e Exemplos

2.5.1 Segmentações Satisfatórias e Insatisfatórias

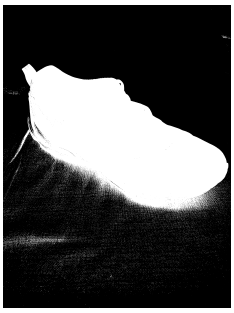

Figura 1: Exemplo de segmentação satisfatória.

Figura 2: Exemplo de segmentação insatisfatória.

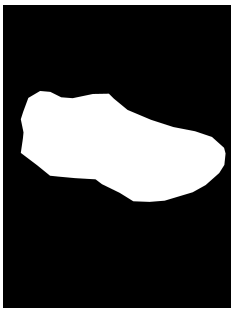

Figura 3: Exemplo de máscara manual (ground-truth).

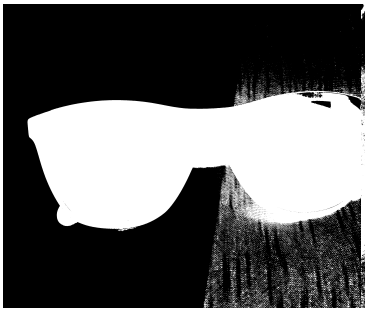

Figura 4: Exemplo de segmentação satisfatória.

Figura 5: Exemplo de segmentação insatisfatória.

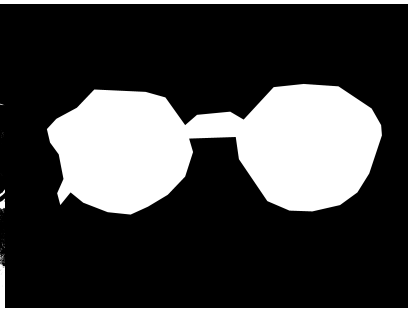

Figura 6: Exemplo de máscara manual (ground-truth).

Figura 7: Exemplo de segmentação satisfatória.

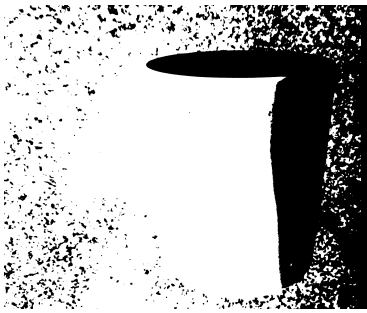

Figura 8: Exemplo de segmentação insatisfatória.

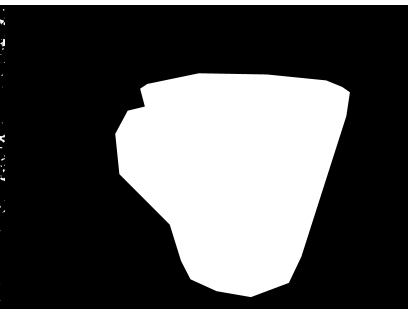

Figura 9: Exemplo de máscara manual (ground-truth).

3 EP3.2: Avaliação do Método de Classificação

3.1 Objetivo

Avaliar o desempenho do método de classificação com base nas imagens segmentadas.

3.2 Metodologia

- Representação das RoIs (Regiões de Interesse) utilizando o Feret Box.
- Descrição das RoIs com PCA (Análise de Componentes Principais).
- Classificação utilizando o algoritmo SVM (Máquina de Vetor de Suporte).
- Divisão do dataset em treino e teste para validação cruzada.

3.3 Resultados e Análise de Performance

Os resultados indicam:

- **Precisão**: 50%
- **Revocação (Recall)**: 50%
- **Acurácia**: 50%
- **F1-Score**: 50%

Tabela 1: Relatório de Classificação por Classe

Classe	Precisão	Revocação	F1-Score	Suporte
Caneca	0.40	0.40	0.40	42
Öculos	0.49	0.44	0.46	41
Relógio	0.51	0.57	0.54	42
Ténis	0.62	0.59	0.60	41

Além disso, foi realizada uma comparação entre diferentes *kernels* para o SVM:

Tabela 2: Comparação de Kernels do SVM

Kernel	Acurácia
Linear	50%
RBF	77.7%
Polinomial	73.5%

3.4 Análise

Os resultados revelaram:

- O kernel RBF apresentou o melhor desempenho, sugerindo maior flexibilidade para capturar relações não lineares.
- Apesar da simplicidade, o kernel linear obteve desempenho significativamente inferior.
- A acurácia geral foi influenciada pelas dificuldades em segmentações complexas.

A matriz de confusão mostra as taxas de acertos e erros entre as classes envolvidas:

Adicionalmente, foi analisado o impacto dos autovetores do PCA na explicação da variância do conjunto de dados. A Figura 11 apresenta os componentes principais mais relevantes, conforme gerados pelo *outputPCAAutovetores*.

3.5 Conclusão

O método apresentou desempenho consistente em imagens com boa segmentação, mas houve queda de performance em casos de segmentações insatisfatórias. Melhorias sugeridas incluem:

- Refinar a qualidade das segmentações para reduzir ruídos.
- Explorar mais componentes no PCA ou utilizar técnicas alternativas para reduzir dimensionalidade.
- Testar outros kernels no SVM, como RBF ou polinomial, para capturar complexidades entre as classes.
- Ampliar o conjunto de dados para melhorar a representatividade.

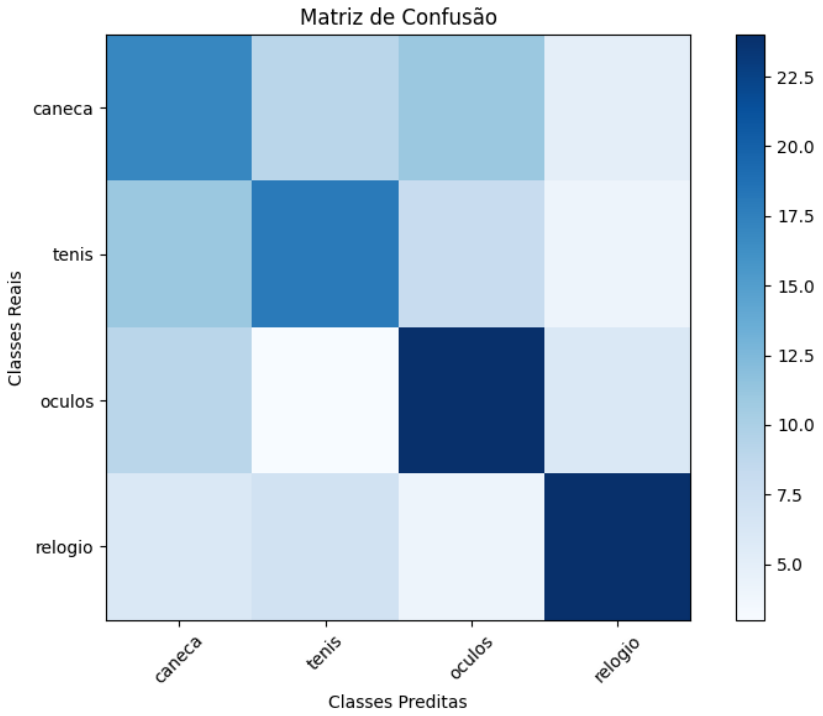

Figura 10: Matriz de confusão mostrando a classificação entre as classes *caneca* e *öculos*.

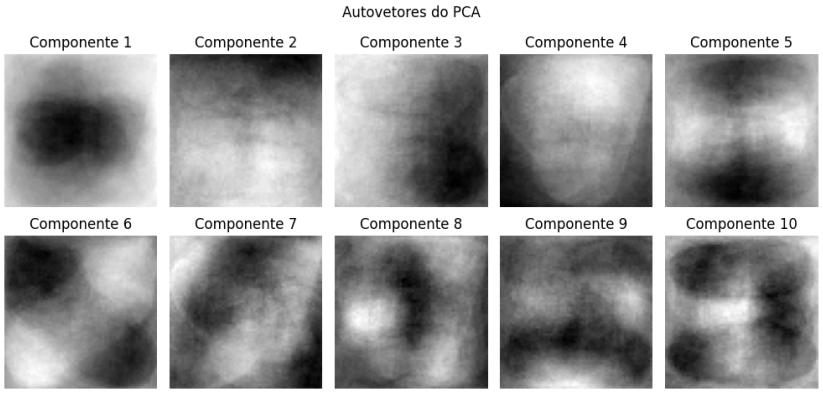

Figura 11: Autovetores mais relevantes gerados pelo PCA.

4 Trabalhos Futuros

Embora este trabalho tenha alcançado seus objetivos principais, existem algumas direções promissoras que podem ser exploradas para aprimorar os resultados em estudos futuros. Entre essas possibilidades, destacam-se:

- **Criação de Conjuntos de Dados Especializados**: O desenvolvimento de conjuntos de dados personalizados e balanceados é fundamental para garantir o treinamento adequado dos modelos. Esses conjuntos poderiam incluir imagens anotadas com maior precisão, abrangendo variações em diferentes condições (iluminação, escala e orientação). Além disso, pode-se considerar a ampliação do dataset com técnicas de aumento de dados (*data augmentation*) para melhorar a generalização do modelo em cenários mais desafiadores.

- **Avaliação de Modelos em Cenários de Tempo Real**: A aplicação em tempo real é um dos desafios mais práticos em projetos de visão computacional. Para alcançar esse objetivo, é necessário otimizar os métodos propostos, reduzindo a latência sem comprometer a precisão. Estratégias como quantização de modelos, uso de bibliotecas otimizadas (como TensorRT) e execução em dispositivos especializados, como GPUs embarcadas ou TPUs, podem ser investigadas.

O aprofundamento nessas áreas permitirá avanços significativos na eficiência e aplicabilidade dos métodos desenvolvidos, aproximando-os de cenários reais e ampliando seu impacto em soluções práticas.