Universidade Federal de Minas Gerais
Departamento de Ciência da Computação

DCC029/868 – Processamento de Imagens Digitais
Prof. Jefersson Alex dos Santos (jefersson@dcc.ufmg.br)
Monitor Caio Cesar Viana da Silva (caiosilva@ufmg.br)

TP2: Classificação de Imagens com Banco de Filtros

1 Descrição

O trabalho DEVERÁ ser desenvolvido em dupla.

O objetivo deste trabalho é a implementação de um programa na linguagem Python, em um notebook Jupyter, para realizar a classificação de imagens a partir do uso de um banco de filtros.

Dentre os diversos problemas da visão computacional, se encontra a classificação de imagens. Dada um conjunto de imagens, é possível criar um algoritmo que ensine o computador a classificar futuras imagens inéditas de forma correta com suas classes. Este problema é bastante trabalhado por diversos pontos de vistas diferentes, envolvendo soluções baseadas desde filtros até mesmo utilização de redes de aprendizado profundo.

Uma forma mais simples, porém que apresenta resultados bastante satisfatórios, é a utilização de um banco de filtros, ou seja, um conjunto de filtros com finalidades diferentes, que quando aplicados as imagens permitam extrair características das mesmas. Estas características aprendidas podem ser utilizadas como comparação com as características extraídas das imagens inéditas. Desta forma, pode-se classificar uma imagem nunca antes vista, de acordo com a similaridade entre suas características e as já conhecidas pelo computador.

Neste trabalho prático, será realizada a criação de um algoritmo que faça a classificação de imagens: a partir de um conjunto de dados para treino, espera-se que o aluno crie um banco de filtros aleatórios, aplique os mesmos às imagens e utilize as características obtidas para comparar com imagens novas.

É esperado também, que sejam utilizados filtros de segunda ordem, ou seja, que as características das imagens sejam obtidas após utilizar a saída de uma das aplicações de um filtro como entrada para uma aplicação de outro filtro.

O banco de filtros deverá ser produzido pelo aluno, sem a utilização de filtros prontos, ou fornecidos por bibliotecas. Além disso, a ideia é que o aluno utilize algum algoritmo evolutivo ou de otimização, para melhorar o banco de filtros a partir dos resultados encontrados nas comparações.

A fim de testar a qualidade dos filtros criados, o aluno deverá separar o conjunto de imagens fornecidos em dois grupos, um grupo de treinamento e um de validação. O grupo de treinamento serão as imagens das quais o computador/algoritmo irá aprender as características, sendo que deverão ser respeitadas as classes de cada uma das imagens. O segundo grupo, de validação, serão as imagens que o aluno irá utilizar pra testar se seu algoritmo está acertando as classificações ou não.

Para avaliar a qualidade do algoritmo desenvolvido, o aluno deverá calcular a acurácia do mesmo. Sendo que a acurácia é a probabilidade do resultado da sua classificação estar correta, sendo determinada pela fórmula:

$$\frac{TP + TN}{TP + FN + FP + TN}$$

Sendo que:
TP = Verdadeiro Positivo (resultado: Positivo, esperado:Positivo)
TN = Verdadeiro Negativo (resultado: Negativo, esperado:Negativo)
FP = Falso Positivo (resultado: Positivo, esperado:Negativo)
FN = Falso Negativo (resultado: Negativo, esperado:Positivo)

A Figura 1 apresenta um resumo do fluxo que deve ser aplicado.

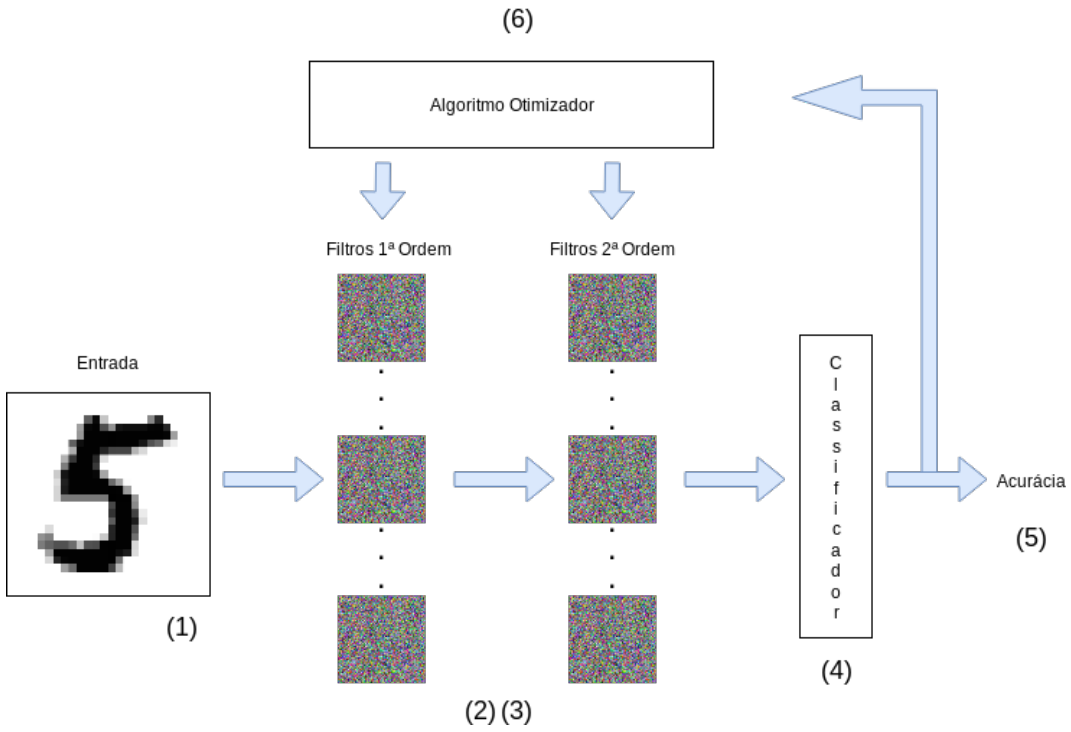

Figura 1: Fluxo de execução do algoritmo. Etapas descritas na Seção 3.

2 Formato da entrada e da saída

Nesse trabalho prático, você deverá implementar um programa que recebe como **entrada** o conjunto de dados de imagens fornecidos, sendo que a divisão de imagens para geração do código e validação fica a escolha do aluno. Como **saída**, o programa deverá exibir no documento os valores de acurácia obtidos para os testes.

3 O que precisa ser feito

Você deverá desenvolver um programa que cumpra os seguintes requisitos:

1. Leitura do conjunto de dados fornecido
2. Geração de um banco de filtros
3. Aplicação do banco de filtros para extração de características
4. Utilização das características para classificação
5. Cálculo da acurácia
6. Atualização dos filtros, para aumento da acurácia obtida

4 O que será fornecido

Será fornecido um conjunto de dados com diversas classes diferentes. O conjunto de dados, MNIST, contém um total de 60000 imagens que representam números de tamanho 28x28, divididas entre 10 classes diferentes.

Cada dupla irá trabalhar com um subconjunto da MNIST. Do conjunto MNIST completo a dupla utilizará apenas duas classes, a primeira é igual ao último dígito da matrícula de

um aluno da dupla e a segunda classe é igual ao último dígito de matrícula do outro aluno. Caso os últimos dígitos sejam iguais, deve-se utilizar o penúltimo dígito, assim por diante.

Exemplo: Caso a dupla seja formada pelas matrículas: 2018664837 e 2018663113, as classes usadas serão 7 e 3.

Cabe ao aluno dividir as imagens fornecidas para realizar o desenvolvimento do código e testar o mesmo.

5 Entrega do código / Documentação

O código e a documentação devem ser entregues em um único arquivo notebook Jupyter (.ipynb). Caso seja utilizado algum arquivo extra, o mesmo deve ser entregue junto ao notebook Jupyter em um arquivo Zip.

O texto da documentação deve ser breve, de forma que o corretor possa entender o que foi feito no código sem ter que entender linha a linha dos arquivos. Implementações modularizadas deverão mencionar quais funções são implementadas em cada módulo ou classe. A documentação deve conter os seguintes itens:

- Sumário do problema a ser tratado.
- Uma descrição sucinta dos algoritmos, das principais funções, e procedimentos.
- Decisões de implementação que porventura estejam omissos na especificação.

O código deve estar inserido ao longo da documentação e deve estar funcional, de forma que a avaliação será realizada ao executar o notebook Jupyter para obter os resultados finais.

Obs.: Programas que não compilarem não serão corrigidos. Trabalhos que forem entregues sem a documentação não serão corrigidos. Trabalhos que forem entregues em algum formato que não notebook Jupyter não serão corrigidos.

6 Avaliação

Esse trabalho prático vale **15 pontos**, sendo os pontos distribuídos para a implementação correta de cada um dos requisitos detalhados na Seção 3. O trabalho pode ser penalizado (obter pontos negativos) se a documentação, clareza do código e funcionamento correto do programa não estiverem de acordo com o requerido.

As regras para desconto de nota por atraso são:

- Atraso de até 24 horas: -1.5 pontos.
- Atraso de 24 horas até 48 horas: -3 pontos.
- Atraso de 48 horas até 72 horas: -4.5 pontos.
- Atraso de mais de 72 horas: -10 pontos.